Noitom Technology Co., Ltd.

PNLib Runtime API Documentation

For PNLib 3.1.30.5581 or latter
May, 9th, 2015

By Yuanhui He, Yu Wang, Peng Gao, Siyuan Deng, Rich Shoda

Noitom Technology Co., Ltd.
www.neuronmocap.com

Contents

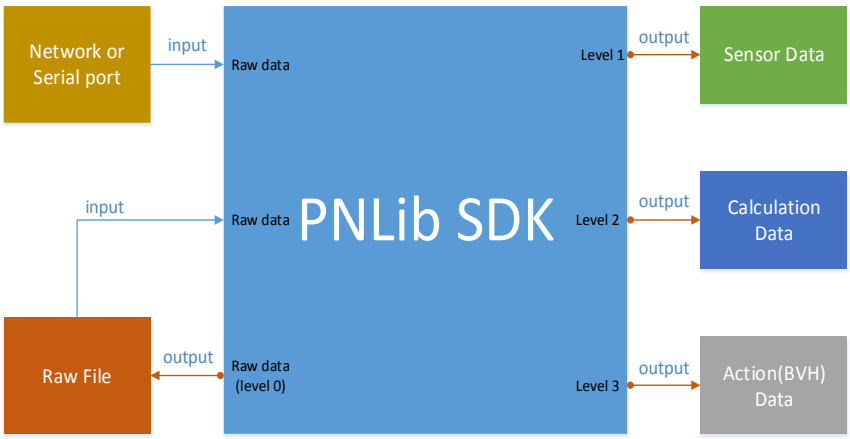
1 Overview

1.1 PNLib SDK framework

PNLib SDK(PNLib Runtime API) integrate a series of functions including reading and analyzing raw data, data fusion, format conversion, motion capture data output, model driven etc. User could achieve complex features by simple configuration of function calling. The coding work is minimal.

According to the classification, there are mainly three types of output data from PNLib SDK regard as three levels:

- Level 1: Sensor data
  For analysis and processing used a single sensor;
- Level 2: Calculation data
  Analysis data for labaray user. Include position, velocity, raw pose quaternion, raw acceleration data raw gyro data of every sensor;
- Level 3: Motion data
  Output is mainly for action data integration of Motion Capture.

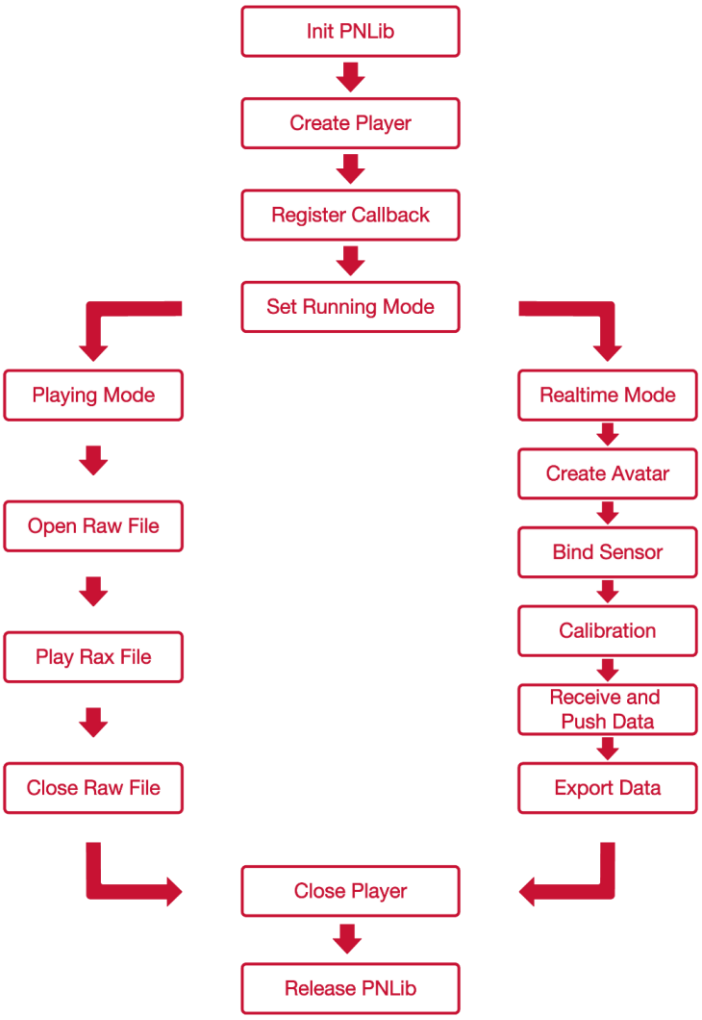


对于动作数据，主要有两个应用方向，即对于数据精度要求极高的动漫领域和对于实时性要求比较高的 VR 领域，PNLib SDK 提供了极其丰富的参数接口已达到各自不同的目的。

PNLib SDK supports many develop languages and interface library, such as C/C++/MFC, WPF/C#, Mac Cocoa, and supports game engines such as Unity, Unreal Engine and Unigine.

1.2 Flowchart of calling PNLib SDK

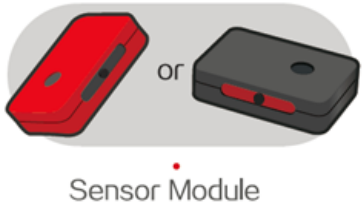
The main steps of calling PNLib contains initing PNlLib, creating player, registering callback needed, recording motion caption data or playing raw file, closing player and releasing PNLib are as below:



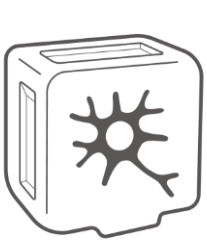
1.3 Sensor

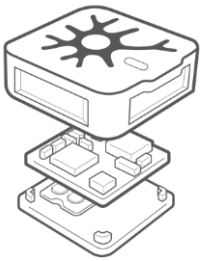
到目前为止，Noitom 提供了两种传感器用于采集数据，一种是面向电影及动漫行业的高精度高质量数据的 Legacy Sensor，另一种是用于高实时性的 VR 领域的 Neuron Sensor：

1.3.1 Legacy Sensor



1.3.2 Neuron Sensor





1.4 Coordinates

1.4.1 Coordinate of Sensor Module

采集模块的坐标轴定义，对于 Legacy，USB 口所在侧面为 X 轴正方向、LED 灯所在侧面为 Z 轴正方向、电源按钮所在侧面为 Y 轴正方向。

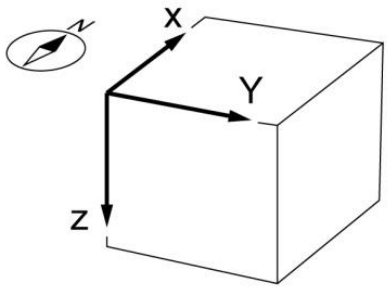
如下图所示：

对于 Neuron，。。。。。。。。。。。。。。。。。。。。。。。。

1.4.2 Coordinate in PNLib

除了输出的 BVH 数据是标准的 BVH 坐标系（’左 X-上 Y-前 Z’）外，整个坐标系都是以’北 X-东 Y-地 Z’作为内部计算坐标系的。

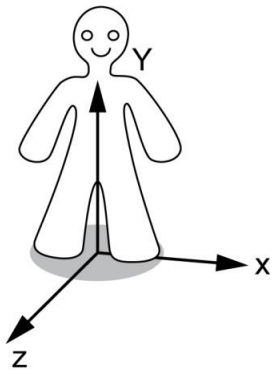
长度单位为米，角度单位为度，重力加速度单位为 g(1g=9.8m/s²)，角速度为度/秒，如下所示：



因此，对于 PNLib 输出的数据，中间数据遵从 PNLib 库的坐标系定义，BVH 数据遵从 BVH 标准定义。

1.4.3 BVH data coordinate definition

PNLib 输出的 BVH 遵守标准 BVH 数据格式定义，Displacement 按左上前的坐标系定义，长度单位为厘米；姿态的定义有六种可选：XYZ, YZX, ZXY, XZY, ZYX, YXZ，系统默认为 YXZ，单位为度。


Y
X
Z

1.5 Skeleton System

There are 167 bones in standard bone table. Legacy can accquire 21 bones' data, and Neuron can accquire 59 bones' data include fingers.

1.5.1 Standard Bone Table

The following table shows the standard bone index and bone name in PNLib. Note that not all bones will binding sensors.

下表为 PNLib 库中使用的标准骨骼系统编码表，包含了骨骼索引和对应的骨骼名称。需要注意的是，在正常使用的时候，不是每根骨骼都有传感器，比如单臂模式，只有上臂、前臂和手掌绑定了传感器。在必要情况下，根据不同的绑定情况，PNLib 会根据情况自动计算需要自动补充的骨骼数据。

BoneIndex	BoneName
0	Hips
1	RightUpLeg
2	RightLeg
3	RightFoot
4	LeftUpLeg
5	LeftLeg
6	LeftFoot
7	RightShoulder
8	RightArm
9	RightForeArm
10	RightHand
11	LeftShoulder
12	LeftArm
13	LeftForeArm
14	LeftHand
15	Head
16	Neck
17	Spine3
18	Spine2
19	Spine1
20	Spine
21	RightToeBase
22	LeftToeBase
23	RightFingerBase
24	RightInHandThumb
25	RightHandThumb1
26	RightHandThumb2
27	RightHandThumb3
28	RightHandThumb4
29	RightInHandIndex
30	RightHandIndex1
31	RightHandIndex2
32	RightHandIndex3
33	RightHandIndex4
34	RightInHandMiddle
35	RightHandMiddle1
36	RightHandMiddle2
37	RightHandMiddle3
38	RightHandMiddle4
39	RightInHandRing
40	RightHandRing1
41	RightHandRing2
42	RightHandRing3
43	RightHandRing4
44	RightInHandPinky
45	RightHandPinky1
46	RightHandPinky2
47	RightHandPinky3
48	RightHandPinky4
49	RightInHandExtraFinger
50	RightHandExtraFinger1
51	RightHandExtraFinger2
52	RightHandExtraFinger3
53	RightHandExtraFinger4
54	LeftFingerBase
55	LeftInHandThumb
56	LeftHandThumb1
57	LeftHandThumb2
58	LeftHandThumb3
59	LeftHandThumb4
60	LeftInHandIndex
61	LeftHandIndex1
62	LeftHandIndex2

63	LeftHandIndex3
64	LeftHandIndex4
65	LeftInHandMiddle
66	LeftHandMiddle1
67	LeftHandMiddle2
68	LeftHandMiddle3
69	LeftHandMiddle4
70	LeftInHandRing
71	LeftHandRing1
72	LeftHandRing2
73	LeftHandRing3
74	LeftHandRing4
75	LeftInHandPinky
76	LeftHandPinky1
77	LeftHandPinky2
78	LeftHandPinky3
79	LeftHandPinky4
80	LeftInHandExtraFinger
81	LeftHandExtraFinger1
82	LeftHandExtraFinger2
83	LeftHandExtraFinger3
84	LeftHandExtraFinger4
85	RightInFootThumb
86	RightFootThumb1
87	RightFootThumb2
88	RightFootThumb3
89	RightFootThumb4
90	RightInFootIndex
91	RightFootIndex1
92	RightFootIndex2
93	RightFootIndex3
94	RightFootIndex4
95	RightInFootMiddle
96	RightFootMiddle1
97	RightFootMiddle2
98	RightFootMiddle3
99	RightFootMiddle4
100	RightInFootRing
101	RightFootRing1
102	RightFootRing2
103	RightFootRing3
104	RightFootRing4
105	RightInFootPinky
106	RightFootPinky1
107	RightFootPinky2
108	RightFootPinky3
109	RightFootPinky4
110	RightInFootExtraFinger
111	RightFootExtraFinger1
112	RightFootExtraFinger2
113	RightFootExtraFinger3
114	RightFootExtraFinger4
115	LeftInFootThumb
116	LeftFootThumb1
117	LeftFootThumb2
118	LeftFootThumb3
119	LeftFootThumb4
120	LeftInFootIndex
121	LeftFootIndex1
122	LeftFootIndex2
123	LeftFootIndex3
124	LeftFootIndex4
125	LeftInFootMiddle
126	LeftFootMiddle1
127	LeftFootMiddle2
128	LeftFootMiddle3
129	LeftFootMiddle4
130	LeftInFootRing
131	LeftFootRing1
132	LeftFootRing2
133	LeftFootRing3
134	LeftFootRing4
135	LeftInFootPinky
136	LeftFootPinky1
137	LeftFootPinky2
138	LeftFootPinky3
139	LeftFootPinky4
140	LeftInFootExtraFinger
141	LeftFootExtraFinger1
142	LeftFootExtraFinger2
143	LeftFootExtraFinger3
144	LeftFootExtraFinger4
145	Neck1
146	Neck2
147	Neck3
148	Neck4

149	Neck5
150	Neck6
151	Neck7
152	Neck8
153	Neck9
154	Spine4
155	Spine5
156	Spine6
157	Spine7

158	Spine8
159	Spine9
160	Props0
161	Props1
162	Props2
163	Props3
164	Props4
165	RightShoulderExtra
166	LeftShoulderExtra

1.5.2 Bone Table of Legacy

Bone table of Legacy suit type with data outputs includes 21 bones as below:

BoneIndex	BoneName
0	Hips
1	RightUpLeg
2	RightLeg
3	RightFoot
4	LeftUpLeg
5	LeftLeg
6	LeftFoot
7	RightShoulder
8	RightArm
9	RightForeArm
10	RightHand
11	LeftShoulder
12	LeftArm
13	LeftForeArm
14	LeftHand
15	Head
16	Neck
17	Spine3
18	Spine2
19	Spine1
20	Spine

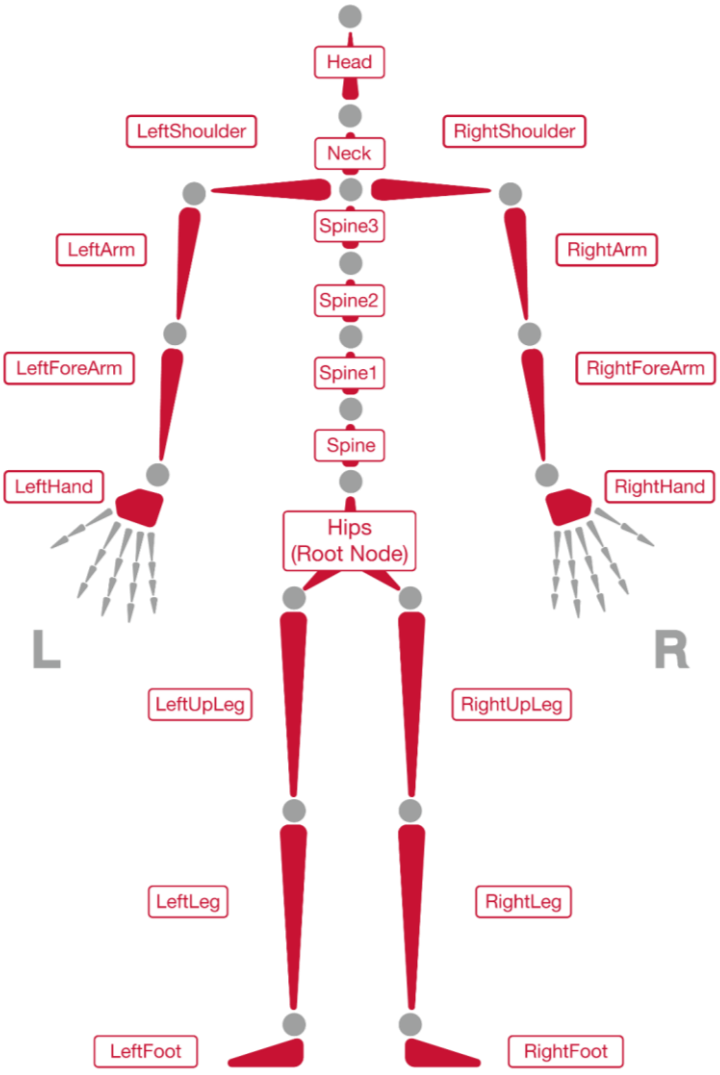
1.5.3 Bone Table of Neuron

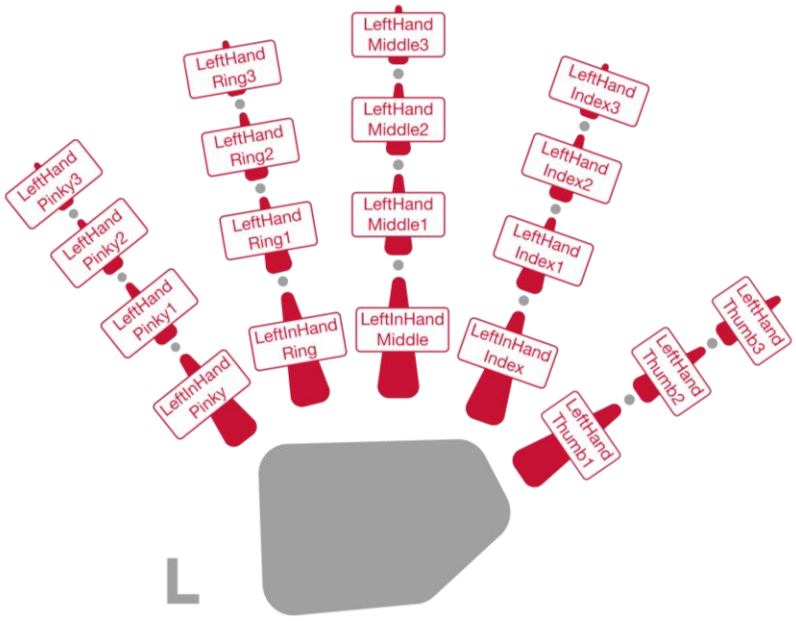
Bone table of Neuron suit type with data outputs includes 59 bones as below:

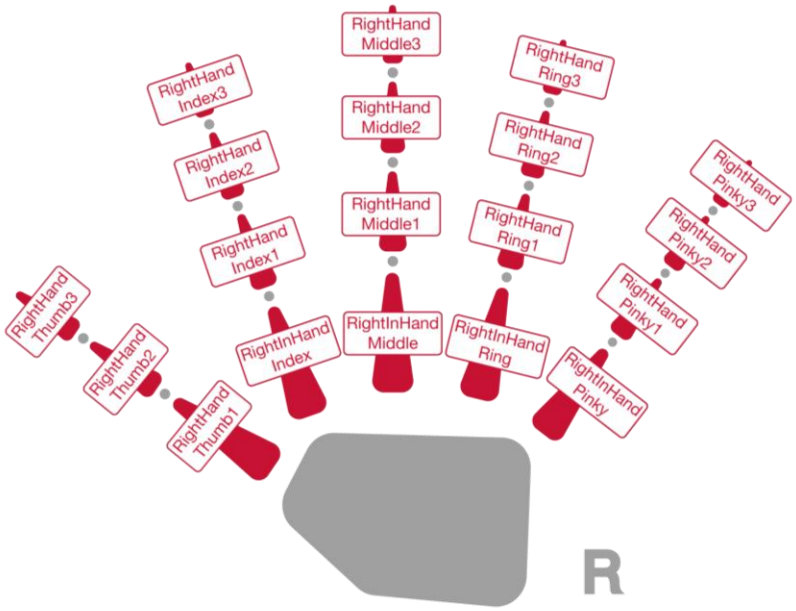
Number	BoneIndex	BoneName
1	0	Hips
2	1	RightUpLeg
3	2	RightLeg
4	3	RightFoot
5	4	LeftUpLeg
6	5	LeftLeg
7	6	LeftFoot
8	7	RightShoulder
9	8	RightArm
10	9	RightForeArm
11	10	RightHand
12	11	LeftShoulder
13	12	LeftArm
14	13	LeftForeArm
15	14	LeftHand
16	15	Head
17	16	Neck
18	17	Spine3
19	18	Spine2
20	19	Spine1
21	20	Spine
22	25	RightHandThumb1
23	26	RightHandThumb2
24	27	RightHandThumb3
25	29	RightInHandIndex
26	30	RightHandIndex1
27	31	RightHandIndex2
28	32	RightHandIndex3
29	34	RightInHandMiddle
30	35	RightHandMiddle1
31	36	RightHandMiddle2
32	37	RightHandMiddle3
33	39	RightInHandRing
34	40	RightHandRing1
35	41	RightHandRing2
36	42	RightHandRing3
37	44	RightInHandPinky
38	45	RightHandPinky1
39	46	RightHandPinky2
40	47	RightHandPinky3
41	56	LeftHandThumb1

42	57	LeftHandThumb2
43	58	LeftHandThumb3
44	60	LeftInHandIndex
45	61	LeftHandIndex1
46	62	LeftHandIndex2
47	63	LeftHandIndex3
48	65	LeftInHandMiddle
49	66	LeftHandMiddle1
50	67	LeftHandMiddle2
51	68	LeftHandMiddle3
52	70	LeftInHandRing
53	71	LeftHandRing1
54	72	LeftHandRing2
55	73	LeftHandRing3
56	75	LeftInHandPinky
57	76	LeftHandPinky1
58	77	LeftHandPinky2
59	78	LeftHandPinky3

1.5.4 Skeleton Nodes




LeftHand Middle3
LeftHand Ring3
LeftHand Index3
LeftHand Middle2
LeftHand Ring2
LeftHand Index2
LeftHand Pinky3
LeftHand Middle1
LeftHand Pinky2
LeftHand Ring1
LeftHand Index1
LeftHand Thumb3
LeftHand Pinky1
LeftInHand Ring
LeftInHand Middle
LeftInHand Index
LeftHand Thumb2
LeftInHand Pinky
LeftHand Thumb1
L


RightHand Middle3
RightHand Index3
RightHand Ring3
RightHand Middle2
RightHand Index2
RightHand Ring2
RightHand Pinky3
RightHand Middle1
RightHand Pinky2
RightHand Index1
RightHand Ring1
RightHand Thumb3
RightHand Pinky1
RightInHand Index
RightInHand Middle
RightInHand Ring
RightHand Thumb2
RightInHand Pinky
RightHand Thumb1
R

1.6 Output data

由于采集数据到达时机的不确定性，PNLib 收集并处理过的数据通过 Callback 的方式回调到用户层。其他配置信息及参数则可通过一些 Get 函数获取到。

1.6.1 Time sequence

对于单 Sensor 数据、中间数据、道具数据、BVH 数据，PNLib 的输出顺序为：

1 单 Sensor 数据
2 中间数据
3 道具数据
4 BVH 数据

处理过程为：当 PNLib 成功解析到某一个 Sensor 数据包后，立即调用单 Sensor 数据输出回调函数，将刚刚分解到的数据传递给用户（PNEventRawDataParsedCallback）；当所有 Sensor 数据收集齐备，则 PNLib 开始计算姿态数据，计算完毕后立即调用 PNEventCalculatedBinaryDataCallback 数据，将中间数据输出给用户；如果用户还激活了道具，则继续计算道具数据并通过道具接口回调道具数据到用户层；最后，将计算结果转换为具有父子结构的 BVH 数据，并将其通过 Callback 的方式回调输出给用户。

1.6.2 Data format

1.7 Prop

PNLib 同样提供对附加道具的支持。如果需要增加道具节点，则只需告诉 PNLib 启用哪个道具（骨骼）节点即可，不需要则将其禁用。

标准骨骼表涵盖了道具模块的定义，Bone ID 为 161~165。

1.8 Node modes

There are 4 kinds of sensor combination modes:

(1)SC_ArmOnly mode:
2 nodes of left upper arm and left fore arm, or, right upper arm and right forearm are necessary.

(2)SC_UpperBody mode:
4 nodes of chest, hips, left upper arm and left forearm, or, chest, hips, right upper arm and right forearm are necessary.

(3)SC_FullBody mode:
6 nodes of left upper leg, left leg, right upper leg, right leg, hips and chest sensor are necessary.

(4)SC_LowerBody mode:
Left upper leg, left leg, right upper leg, right leg, hips and chest

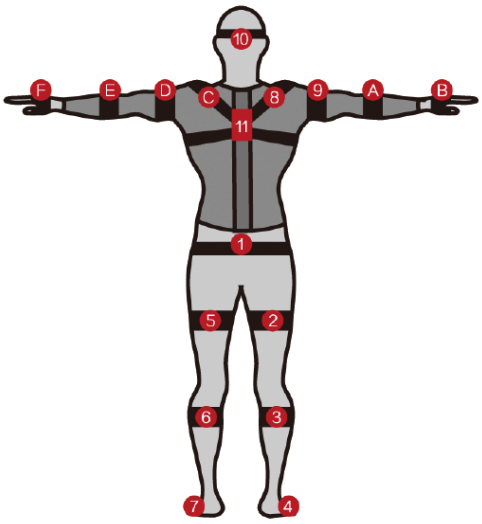
sensor are necessary. Totally 6 nodes.

1.9 Diagram of wear

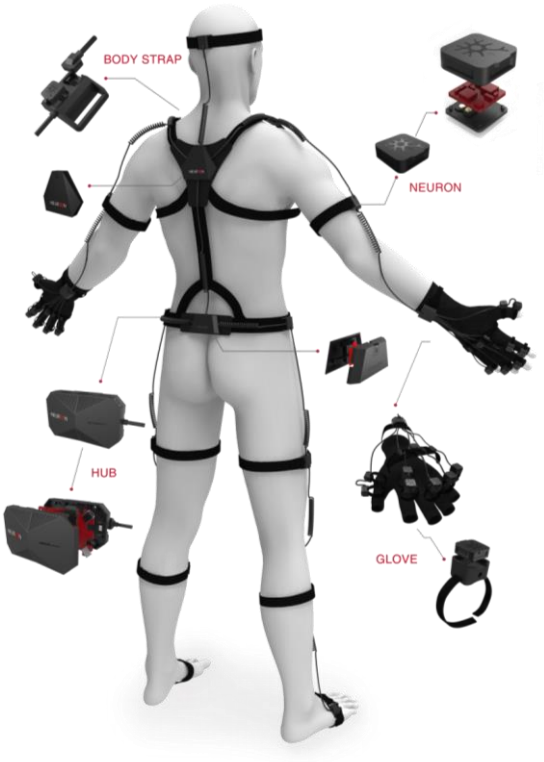
(1) Legacy

Bone table and position of sensor bound on body:

Hips	01
Right Upper Leg	02
Right Leg	03
Right Foot	04
Left Upper Leg	05
Left Leg	06
Left Foot	07
Right Shoulder	08
Right Arm	09
Right Forearm	0A
Right Hand	0B
Left Shoulder	0C
Left Arm	0D
Left Forearm	0E
Head	10
Vertebra	11



(2) Neuron



2 Reference

2.1 Data type definitions

2.1.1 SensorSuitTypes

Sensor suit types at realtime mode.

```
typedef enum _SensorSuitTypes
{
    SS_LegacySensors,
    SS_NeuronSensors,
    SS_Unknown,
}SensorSuitTypes;
```

Members

SS_LegacySensors
Wireless legacy suit, supports 17 sensor modules at most.

SS_NeuronSensors
Neuron Sensors suit with TCP/IP.

SS_Unknown
Unknown type.

2.1.2 SensorCombinationModes

Sensor combination mode.

```
typedef enum _SensorCombinationModes
{
    SC_ArmOnly,
    SC_UpperBody,
    SC_FullBody,
    SC_Unknown,
}SensorCombinationModes;
```

Members

SC_ArmOnly
2 nodes of left upper arm and left forearm are necessary, or, right upper arm and right forearm are necessary.

SC_UpperBody
4 nodes of chest, hips, left upper arm and left forearm, or, chest, hips, right upper arm and right forearm are necessary.

SC_FullBody
6 nodes of left upper leg, left leg, right upper leg, right leg, hips and chest sensor are necessary.

SC_Unknown
Unknown mode.

2.1.3 RunningMode

Running mode of PNLib.

```
typedef enum _RunningMode
{
    RM_Realtime,
    RM_RawPlaying,
    RM_Unknown,
}RunningMode;
```

Members

RM_Realtime
Play in the window while record a motion capture.

RM_RawPlaying
Play a raw file which was recorded a motion capture.

RM_Unknown
Unknown mode.

2.1.4 SensorAcceleratorTypes

Accelerator type of sensor.

```
typedef enum _SensorAcceleratorTypes
{
    SA_Range8G,
    SA_Range16G,
    SA_Range24G,
    SA_Unknown,
}SensorAcceleratorTypes;
```

Members

SA_Range8G
Small range of accelerator type: 8G

SA_Range16G
Medium range of accelerator type: 16G

SA_Range24G,
Large range of accelerator type: 24G

SA_Unknown
Unknown type

2.1.5 RotateOrders

Output order of rotation values around axes in BVH data.

```
typedef enum _RotateOrders
{
    RO_XZY,
    RO_YXZ,
    RO_XYZ,
    RO_YZX,
```

```
    RO_ZXY,
    RO_ZYX,
    RO_Unknown,
}RotateOrders;
```

Members

RO_XZY
Rotation value in X axis first, then Z axis, then Y axis.

RO_YXZ
Rotation value in Y axis first, then X axis, then Z axis.

RO_XYZ
Rotation value in X axis first, then Y axis, then Z axis.

RO_YZX
Rotation value in Y axis first, then Z axis, then X axis.

RO_ZXY
Rotation value in Z axis first, then X axis, then Y axis.

RO_ZYX
Rotation value in Z axis first, then Y axis, then X axis.

RO_Unknown
Unknown type.

2.1.6 OutputQuaternionTypes

Quaternion types of output stream.

```
typedef enum _OutputQuaternionTypes
{
    QT_GlobalRawQuat,
    QT_GlobalBoneQuat,
    QT_LocalBoneQuat,
    QT_Unknown,
}OutputQuaternionTypes;
```

Members

QT_GlobalRawQuat
Module quaternion in the world coordinate system.

QT_GlobalBoneQuat
Calibrated module quaternion in the world coordinate system.

QT_LocalBoneQuat
Quaternion relative to parent node.

QT_Unknown,
Unknown type.

2.1.7 OutputAccelerationTypes

Acceleration data types of output stream.

```
typedef enum _OutputAccelerationTypes
{
```

```
    AT_ModuleRawData,
    AT_GlobalData,
    AT_Unknown,
}OutputAccelerationTypes;
```

Members

AT_ModuleRawData
Module Acceleration in the world coordinate system.

AT_GlobalData
Calibrated module acceleration in the world coordinate system.

AT_Unknown
Unknown type.

2.1.8 OutputGyroType

Gyro data types of output stream.

```
typedef enum _OutputGyroType
{
    GY_ModuleRawData,
    GY_GlobalData,
    GY_Unknown,
}OutputGyroType;
```

Members

GY_ModuleRawData
Module angular velocity in global system.

GY_GlobalData
Calibrated module angular velocity (Bone angular velocity).

GY_Unknown
Unknown type.

2.1.9 BvhDataStreamTypes

Data types of BVH output stream.

```
typedef enum _BvhDataStreamTypes
{
    BO_BinaryType,
    BO_StringType,
    BO_MatrixStringType,
    BO_Unknown,
}BvhDataStreamTypes;
```

Members

BO_BinaryType
Binary type.

BO_StringType
String type.

BO_MatrixStringType

Matrix string type.

BO_Unknown

Unknown type.

2.1.10 CalculatedDataStreamTypes

Types of calculated data stream.

```
typedef enum _CalculatedDataStreamTypes
{
    CS_BinaryType,
    CS_StringType,
    CS_Unknown,
}CalculatedDataStreamTypes;
```

Members

CS_BinaryType

Binary type.

CS_StringType

String type.

CS_Unknown

Unknown type.

2.1.11 ConstraintPoint

Bones can be edited as constraint points.

```
typedef enum _ConstraintPoint
{
    CP_Hip        = 0,
    CP_RightFoot   = 3,
    CP_LeftFoot    = 6,
    CP_RightHand   = 10,
    CP_LeftHand    = 14,
    CP_Unknown,
}ConstraintPoint;
```

Members

CP_Hip

Hips

CP_RightFoot

Right foot

CP_LeftFoot

Left foot

CP_RightHand

Right hand

CP_LeftHand

Left hand

CP_Unknown

Unknown type

2.1.12 MagneticImmunityLevel

Magnetic immunity levels.

```
typedef enum _MagneticImmunityLevel
{
    MI_Disable,
    MI_Weak,
    MI_Strong,
    MI_Unknown,
}MagneticImmunityLevel;
```

Members

MI_Disable
Disable magnetic immunity function.

MI_Weak
Low level magnetic immunity.

MI_Strong
Strong level magnetic immunity.

MI_Unknown
Unknown type.

2.1.13 CalibrationTypes

Poses for calibration.

```
typedef enum _CalibrationTypes
{
    Cali_TPose,
    Cali_APose,
    Cali_Spose,
    Cali_NPose,
    Cali_Unknown,
}CalibrationTypes;
```

Members

Cali_Tpose
T pose

Cali_Apose
A pose

Cali_Spose
Crouching pose

Cali_Npose
Nod pose

Cali_Unknown
Unknown type

2.1.14 PNLibVersion

Dynamic library version info.

```
typedef struct _PNLibVersion
{
    USHORT Major;
    USHORT Minor;
    USHORT Revision;
    USHORT BuildNumb;
}PNLibVersion;
```

Members

USHORT Major
Major number

USHORT Minor
Minor number

USHORT Revision
Revision number

USHORT BuildNumb
Build number

2.1.15 OutputDataVersion

BVH data stream version info.

```
typedef union _OutputDataVersion
{
    UINT32 _VersionMask;
    struct
    {
        UCHAR BuildNumb;
        UCHAR Revision;
        UCHAR Minor;
        UCHAR Major;
    };
} DATA_VER;
```

Members

_VersionMask
Mask of version

BuildNumb
Build number

Revision
Revision number

Minor
Subversion number

Major
Major version number

2.1.16 BvhOutputBinaryHeader

Header format of BVH data in output stream.

```
typedef struct _BvhOutputBinaryHeader
{
    UINT16 BvhHeaderToken1;
    BVH_DATA_VER DataVersion;
    UINT32 DataCount;
    BOOL WithDisp;
    BOOL WithReference;
    UINT32 AvatarIndex;
    UCHAR AvatarName[32];
    UINT32 Reserved1;
    UINT32 Reserved2;
    UINT16 BvhHeaderToken2;
}BvhOutputBinaryHeader;
```

Members

BvhHeaderToken1
Start token of package: 0xDDFF.
DataVersion
Version of community data format. e.g.: 1.0.0.2.
DataCount
Values count:
WithDisp
With/without displacement.
WithReference
With/without reference bone data at first.
AvatarIndex
Avatar index.
AvatarName
Avatar name.
Reserved1
Reserved, padding bit for 64 bytes length of package.
Reserved2
Reserved, padding bit for 64 bytes length of package.
BvhHeaderToken2
End token of package: 0xEEFF.

2.1.17 BvhOutputBinaryHeaderEx

Header format of compressed BVH data.

```
typedef struct _BvhDataHeaderEx
```

```
{
    UINT16   BvhHeaderToken1;
    DATA_VER DataVersion;
    UINT32   DataCount;
    BOOL     WithDisp;
    BOOL     WithReference;
    UINT32   AvatarIndex;
    UCHAR    AvatarName[32];
    UINT32   IsCompressed;
    UINT32   Reserved1;
    UINT16   BvhHeaderToken2;
}BvhOutputBinaryHeaderEx;
```

Members

BvhHeaderToken1
Start token of package: 0xDDFF.

DataVersion
Version of community data format. e.g.: 1.0.0.2.

DataCount
Different values count with last frame if compressed.

WithDisp
With/without displacement.

WithReference
With/without reference bone data at first.

AvatarIndex
Avatar index.

AvatarName
Avatar name.

IsCompressed
Wether BVH data is compressed and compressed data check code.

Reserved1
Description of same values with last frame if compressed.

BvhHeaderToken2
End token of package: 0xEEFF.

2.1.18 CalculationDataHeader

Header format of calculation data.

```
typedef struct _CalculationDataHeader
{
    UINT16 HeaderToken1;
    DATA_VER DataVersion;
    UINT32 DataCount;
    UINT32 AvatarIndex;
    UCHAR AvatarName[32];
```

```
    UINT32 Reserved1;
    UINT32 Reserved2;
    UINT32 Reserved3;
    UINT32 Reserved4;
    UINT16 HeaderToken2;
}CalculationDataHeader;
```

Members

UINT16 HeaderToken1
Start token of package: 0xDDFF.

DATA_VER DataVersion
Version of community data format. e.g.: 1.0.0.2.

UINT32 DataCount
Values count. 17*(16 floats) for Legacy, 59*(16 floats) for Neuron.

UINT32 AvatarIndex
Avatar index.

UCHAR AvatarName[32]
Avatar name.

UINT32 Reserved1
Reserved, padding bit for 64 bytes length of package.

UINT32 Reserved2
Reserved, padding bit for 64 bytes length of package.

UINT32 Reserved3
Reserved, padding bit for 64 bytes length of package.

UINT32 Reserved4
Reserved, padding bit for 64 bytes length of package.

UINT16 HeaderToken2
End token of package: 0xEEFF.

2.1.19 Quaternion4_t

Quaternion

```
typedef struct _Quaternion4_t
{
    float s;
    float x;
    float y;
    float z;
}Quaternion4_t;
```

Members

s
Rotation angle around rotation axis.

x
X vector to describe rotation axis.

y
Y vector to describe rotation axis.
z
Z vector to describe rotation axis.

2.1.20 Vector3_t

3D Vector with 3 float variable.

```
typedef struct _Vector3_t
{
    float x;
    float y;
    float z;
}Vector3_t;
```

Members

x
Coordinate in X axis.
y
Coordinate in Y axis.
z
Coordinate in Z axis.

2.1.21 BoneMap

Standard bone system table.

```
typedef struct _BoneMap
{
    int Index;
    char Name[32];
    int SensorId;
}BoneMap;
```

Members

Index
Bone index.
Name
Bone name.
SensorId
Sensor id bound to this bone.

2.1.22 BoneDimension

Dimensions of bone, unit: meter

```
typedef struct _BoneDimension
{
    float Head;
```

```
    float Neck;
    float Body;
    float ShoulderWidth;
    float UpperArm;
    float Forearm;
    float Palm;
    float HipWidth;
    float UpperLeg;
    float LowerLeg;
    float HeelHeight;
    float FootLength;
}BoneDimension;
```

Members

Head
Bone length of head, default: 0.18

Neck
Bone length of neck, default: 0.09

Body
Length of body, default: 0.65

ShoulderWidth
Width of shoulder, default: 0.35

UpperArm
Bone length of upper arm, default: 0.29

Forearm
Bone length of fore arm, default: 0.28

Palm
Bone length of hand, default: 0.19

HipWidth
Width of hips, default: 0.23

UpperLeg
Bone length of upper leg, default: 0.48

LowerLeg
Bone length of lower leg, default: 0.48

HeelHeight
Heel height, default: 0.05

FootLength
Foot length, default: 0.28

2.1.23 ContactStatus

Status of constraint point.

```
typedef struct _ContactStatus
{
    ConstraintPoint Point;
```

```
    PNBOOL IsEdited;
    PNBOOL IsContact;
}ContactStatus;
```

Members

Point
Constraint point.

IsEdited
The tag whether it has been edited.

IsContact
The tag whether it is contacting.

2.1.24 FrameContactData

All status of constraint points in one frame.

```
typedef struct _FrameContactData
{
    int FrameIndex;
    ContactStatus ContactInfo[5];
}FrameContactData;
```

Members

FrameIndex
Index of frame data.

ContactInfo
Status of constraints.

2.1.25 RawFileTime

Created time of raw file.

```
typedef struct _RawFileTime
{
    UCHAR Reserved;
    UCHAR Second;
    UCHAR Minute;
    UCHAR Hour;
    UCHAR Day;
    UCHAR Month;
    USHORT Year;
}RawFileTime;
```

Members

Reserved
Millisecond (0~99).

Second
The second while raw file is created.

Minute

The minute while raw file is created.

Hour

The hour while raw file is created.

Day

The day while raw file is created.

Month

The month while raw file is created.

Year

The year while raw file is created.

2.1.26 RawFileAvatarInfo

One avatar information in raw file.

```
typedef struct _RawFileAvatarInfo
{
    char AvatarName[32];
    SensorCombinationModes CombMode;
    float FrontDirection[3];
    BoneDimension boneDimension;
    PNBOOL SensorBindingList[FULL_BODY_BONE_COUNT];
}RawFileAvatarInfo;
```

Members

AvatarName

Avatar's name.

CombMode

Combination modes.

FrontDirection

Front direction.

boneDimension

Bone dimension.

SensorBindingList

Sensor binding list.

2.1.27 RawFileInfo

Information of raw file.

```
typedef struct _RawFileInfo
{
    RawFileTime DateTime;
    PNLibVersion LibVersion;
    SensorSuitTypes SuitType;
    int DataFrequency;
    int TotalFrames;
    int TotalTime;
    int AvatarCount;
```

```
    RawFileAvatarInfo* AvatarInfoList;
}RawFileInfo;
```

Members

DateTime
Created time of raw file.

LibVersion
Library version of PNLib that record this file.

SuitType
Sensor suit type: Neuron or Legacy.

DataFrequency
Data acquisition frequency, Hz.

TotalFrames
Total frames in raw file.

TotalTime
Total time of playing this file, unit:second.

AvatarCount
Avatar count.

AvatarInfoList
Information list of all avatars in raw file.

2.1.28 CalibrationData

Calibration data.

```
typedef struct _CalibrationData
{
    int AvatarIndex;
    char AvatarName[64];
    BoneDimension BoneDim;
    Vector3_t FaceDirection;
    Vector3_t LeftDirection;
    Vector3_t BoneDirections[FULL_BODY_BONE_COUNT];
    Vector3_t BoneLeft[FULL_BODY_BONE_COUNT];
    Vector3_t AccData[FULL_BODY_BONE_COUNT];
}CalibrationData;
```

Members

AvatarIndex
Avatar index in avatar list.

AvatarName
Avatar name.

BoneDim
Bone dimension of avatar.

FaceDirection
Initial face direction at calibrated time.

LeftDirection

```
        Initial left direction at calibrated time.
    BoneDirections
        Each bone direction list.
    BoneLeft
        Each bone left direction list.
    AccData
        Bone acceleration data of calibration.
```

2.1.29 SmoothFactors

Smooth factors.

```
typedef struct _SmoothFactors
{
    int GlobalDisplacement;
    int GlobalRotation;
    int HipDisplacement;
    int FeetDisplacement;
    int HipRotation;
    int FeetRotation;
}SmoothFactors;
```

Members

```
    GlobalDisplacement
         Smooth factor of displacement for all bones.
    GlobalRotation
         Smooth factor of rotation for all bones.
    HipDisplacement
         Smooth factor of displacement special for hips bone, global value
would be overrided by this value.
    FeetDisplacement
         Smooth factor of displacement special for feet bones, global
value would be overrided by this value.
    HipRotation
         Smooth factor of rotation special for hips bone, global value
would be overrided by this value.
    FeetRotation
         Smooth factor of rotation special for feet bones, global value
would be overrided by this value.
```

2.2 Callbacks

2.2.1 PNEventCalibrationProgressCallback

Calibration progress callback.

```
typedef void (__stdcall *PNEventCalibrationProgressCallback)
                  (void* customObject, int avatarIndex, float percent);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

percent

Percentage of avatar current calibration progress; output parameter from PNLib library.

Remarks

Fill body of this function if calibration progress is needed.It will be called once each avatar each frame. Percent equals to 1.0 when calibraton completes.

2.2.2 PNEventPlayProgressCallback

Raw file playing callback.

```
typedef void (__stdcall *PNEventPlayProgressCallback)
              (void* customObject, int currentFrame, int totalFrames);
```

Members

customObject

User defining type.

currentFrame

Current frame number of playing; output parameter from PNLib library.

totalFrames

Total frames of raw file; output parameter from PNLib library.

Remarks

Fill body of this function if playing status of the file is needed. It will be called once each frame. The currentFrame equals to totalFrames subtracting 1 when playing completes.

2.2.3 PNEventRawDataParsedCallback

Parsed raw data callback.

```
typedef void (__stdcall *PNEventRawDataParsedCallback)
              (void* customObject, int sensorId, Quaternion4_t* quat,
              Vector3_t* acc, Vector3_t* gyro);
```

Members

customObject

User defining type.

sensorId

Sensor’s index.

quat

Quaternion corresponding the sensor: Output parameter from PNLib library with four values s, x, y, z.

acc

Acceleration corresponding the sensor: Output parameter from PNLib library with three values x, y, z.

gyro

Gyro corresponding the sensor: Output parameter from PNLib library with three values x, y, z.

Remarks

Fill body of this function if values including quaternion, gyro and acceleration corresponding one sensor are needed.

2.2.4 PNEventCalculatedStringDataCallback

Calculated frame string data callback.

```
typedef void (__stdcall *PNEventCalculatedStringDataCallback)
                          (void* customObject, char* calculationData);
```

Members

customObject

User defining type.

calculationData

Calculated data of current frame: Output parameter from PNLib library.

Remarks

Fill body of this function if calculated data is needed. Format of calculation data refer to its definition.

2.2.5 PNEventCalculatedBinaryDataCallback

Calculated frame binary data callback.

```
typedef void (__stdcall *PNEventCalculatedBinaryDataCallback)
                              (void* customObject, int avatarIndex,
                              CalculationDataHeader* cbp, int packLen);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

cbp

Calculation data header pointer.

packLen

Pack length.

Remarks

Fill body of this function if calculation data is needed as binary type. It will be called once each avatar each frame. After CalculationDataHeader pointer, there are 16 calculation data of float type for each bone as below:
1.Position in global coordinate with X, Y, Z, Unit: meter;
2.Velocity in global coordinate with X, Y, Z, Unit: meter;
3.Rotation described by quaternion and configed by PNSetCalculatedQuaternionDataType function;
4.Acceleration with X, Y, Z and configed by PNSetCalculatedAccelerationDataType function, Unit: gramme;
5.Gyro with X, Y, Z and configed by PNSetCalculatedGyroDataType function, Unit: radian per second.

Data number of Legacy suit type is 16 multiplying 21 bones;
And 16 multiplying 59 bones in Neuron suit type.
Sample of output data refer to appendence.

2.2.6 PNEventBVHStringDataBoardcastCallback

BVH string data broadcast callback.

```
typedef void (__stdcall *PNEventBVHStringDataBoardcastCallback)
                  (void* customObject, int avatarIndex, char* bvhData);
```

Members

customObject
User defining type.

avatarIndex
Avatar index.

bvhData
Bvh data of current frame providing as string type.

Remarks

Set BvhDataStreamTypes to BO_StringType and fill body of this function if bvh data as string type is needed; bvh data format refer to its definition.

2.2.7 PNEventBVHBinaryDataBoardcastCallback

BVH binary data broadcast callback.

```
typedef void (__stdcall *PNEventBVHBinaryDataBoardcastCallback)
                            (void* customObject, int avatarIndex,
                            BvhOutputBinaryHeader * bbp, int packLen);
```

Members

customObject
User defining type.

avatarIndex

Avatar index.

bbp

Pack pointer of BvhOutputBinaryHeader type.

packLen

Pack length which is: dataCount * sizeof(float) + sizeof(BvhOutputBinaryHeader).

Remarks

Set BvhDataStreamTypes to BO_BinaryType and fill body of this function if bvh data as binary type is needed. It will be called once each avatar each frame. After BvhOutputBinaryHeader pointer, there are BVH data of float type for each bone as below:

1.Position of Hips with 3 float data in global coordinate, Unit: centimeter;

2.Rotation of 59 bones with 3 float data in global coordinate;

Rotation data of finger bones will be 0 in Legacy suit type. Whether output Position data of other bones and rotation XYZ output order can be configed by PNSetBvhDataFormat function. Whether output with reference of 6 float data can be configed by PNSetBvhDataWithReference function. So the total data number is:

180 float data array without prefix nor displacements,59*3+3;

186 float data array with prefix and without displacements,59*3+6;

354 float data array without prefix and with displacements,59*6;

360 float data array with prefix and displacements,59*6+6.

Sample of output data refer to appendence.

2.2.8 PNEventBVHMatrixDataBoardcastCallback

BVH matrix string data callback.

```
typedef void (__stdcall *PNEventBVHMatrixDataBoardcastCallback)
              (void* customObject, int avatarIndex, char* matrixData);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

matrixData

Bvn matrix string data of current frame providing as string type.

Remarks

Set BvhDataStreamTypes to BO_MatrixStringType and fill body of this function if bvh data as matrix string type is needed; matrix string data format refer to its definition.

2.2.9 PNEventConstraintDataCallback

Constraint data callback.

```
typedef void (__stdcall *PNEventConstraintDataCallback)
  (void* customObject, int avatarIndex, FrameContactData* contactData);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

contactData

Pointer of FrameContactData type.

Remarks

Fill body of this function if constraint status of an avatar is needed. It will be called once if some contact point of some frame of some avatar is edited. Playing raw file or exporting data also can call this function.

2.2.10 PNEventActionRecognitionDataStringStreamCallback

Action recognition string data callback.

```
typedef void(__stdcall
              *PNEventActionRecognitionDataStringStreamCallback)
             (void* customObject, int avatarIndex, char* actionData);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

actionData

Pointer of action data as string type.

Remarks

Fill body of this function if action recognition data as string type of an avatar is needed. It will be called once each avatarIndex each recognition event. actionData's format is below like that:

R P2 26 1.609 154.501 -58.319 -0.279 -0.759 0.588||

批注 [amy1]: 差动作识别的位移单位和坐标系

Recognition Event	Event Code	Left/Right Hand	Intensity Level	Position of hand	Direction of hand	End Point
Single tap	P0	L	1-10	XYZ	XYZ	||
		R	1-10	XYZ	XYZ	||
Double tap	P1	L	1-10	XYZ	XYZ	||
		R	1-10	XYZ	XYZ	||
Fire	P2	L	1-10	XYZ	XYZ	||
		R	1-10	XYZ	XYZ	||

in-ward flipping	P3	L	1-10	XYZ	XYZ	||
		R	1-10	XYZ	XYZ	||
out-ward flipping	P4	L	1-10	XYZ	XYZ	||
		R	1-10	XYZ	XYZ	||
single clap	P5	N/A	1-10	XYZ	N/A	||
double clap	P6	N/A	1-10	XYZ	N/A	||

2.2.11 PNEventBodyMassVectorStringCallback

Action recognition: body mass vector.

```
typedef void(__stdcall *PNEventBodyMassVectorStringCallback)
                    (void* customObject, int avatarIndex, char* data);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

data

Pointer of body mass data as string vector type.

Remarks

Fill body of this function if body mass data of an avatar as string vector type is needed.

2.2.12 PNEventBodySwingVectorStringCallback

Action recognition: body swing vector.

```
typedef void(__stdcall *PNEventBodySwingVectorStringCallback)
                    (void* customObject, int avatarIndex, char* data);
```

Members

customObject

User defining type.

avatarIndex

Avatar index.

data

Pointer of body swing data as string vector type.

Remarks

Fill body of this function if body swing data as string vector type of an avatar is needed.

2.3 API reference

```
/*********************************************************
```

```
*         Initialize and config PNLib environment      *
*******************************************************/
```

2.3.1 PNGetLibVersion

Get this running PNLib version number.

```
PNLIB_API PNLibVersion PNGetLibVersion();
```

Return Value

Return a structure variable of PNLib library version. Its format is declared in “PNDataTypes.h”.

Remarks

PNLibVersion structure include four USHORT which is defined unsigned short variable type.

2.3.2 PNLibInit

Initialize library.

```
PNLIB_API void PNLibInit();
```

Remarks

Must initialize library environment by calling this function before using PNLib.

2.3.3 PNGetLastErrorCode

Get error code to find more information if calling a function failed.

```
PNLIB_API const PNSTATUS PNGetLastErrorCode();
```

Return Value

This function returns a PNLib error code. PNSTATUS is predefined as unsigned int. '0' means implementing successfully, otherwise certain error occurred when calling PNLib function.

Remarks

Can get last error code of PNLib.

2.3.4 PNGetLastErrorMessage

Get last error information with windows system error code.

```
PNLIB_API const char* PNGetLastErrorMessage();
```

Return Value

Function returns a string message and windows system error code of PNLib last error.

Remarks

Can get last error message and windows system error code if calling a function failed.

2.3.5 PNSetSensorSuitType

Set sensor suit type: Neuron or Legacy sensor type

```
PNLIB_API void PNSetSensorSuitType(SensorSuitTypes modes);
```

Parameters

modes

There are two sensor suit types: Legacy and Neuron.

Remarks

Must set this property before motion capture.

2.3.6 PNGetStandardBoneTable

Get standard bone system table of PNLib.

```
PNLIB_API const BoneMap* PNGetStandardBoneTable();
```

Return Value

Return a array point of BoneMap, the count is defined by 'FULL_BODY_BONE_COUNT'

2.3.7 PNSetSensorCombinationMode

Set sensor combination mode.

```
PNLIB_API PNBOOL PNSetSensorCombinationMode(int avatarIndex,
SensorCombinationModes mode);
```

Return Value

Return TRUE if set successfully, otherwise return FALSE.

Parameters

avatarIndex

Avatar index.

mode

There are three sensor combination modes: Arm only, Upper body and Full body.

Remarks

Must set this property before motion capture. PNCheckSensorBindingMode will be called in this function to check current sensor binding if it is compatible with refered mode.

2.3.8 PNSetDataFolders

Set temporary folder or data export default folder.

```
PNLIB_API void PNSetDataFolders(char* appDataFolder, char* workingFolder);
```

Parameters

appDataFolder

Calibration data will save to appData folder.

workingFolder

Exported data file will be saved to 'workingFolder'.English path is better. Exporting fbx file to a non-latin language path will fail.

Remarks

Calling this function will save the path in PNLib only. It will not create the folder if there isn't.

Example

```
PNSetDataFolders("appDataFolder", "workingFolder");
_mkdir("appDataFolder");
_mkdir("workingFolder");
char* fbxname = PNExportFbxData(avatarIndex);
PNRawDataPlayStart();
Sleep(milliseconds);
PNStopExportFbxData(avatarIndex);
PNRawDataPlayStop();
```

2.3.9 PNSetDataAcquisitionFrequency

Set data acquisition frequency. Unit: Hz

```
PNLIB_API void PNSetDataAcquisitionFrequency(int freq);
```

Parameters

freq

Data acquisition frequency (unit: Hz).

Remarks

Adjust acquisition with this function befor starting to acquire data. Default value is 96Hz in PNLib.

2.3.10 PNGetDataAcquisitionFrequency

Get current data acquisition frequency in PNLib.

```
PNLIB_API int PNGetDataAcquisitionFrequency();
```

Return Value

This function returns current data acquisition frequency.

Remarks

Acquisition frequency depends on different hardware equipment.When capturing motion it will save in raw file.

2.3.11 PNSetRunningMode

Set or switch run mode between real time mode and raw file playing mode.

```
PNLIB_API void PNSetRunningMode(RunningMode runMode);
```

Parameters

runMode

If PNLib runs as real time capture mode, set RM_Realtim. Or just playing raw file, set RM_RawPlaying.

Remarks

Must set this property to rawplaying mode before opening a raw file.

2.3.12 PNGetRunningMode

Get current running mode.

```
PNLIB_API RunningMode PNGetRunningMode();
```

Return Value

This function returns current running mode.

Remarks

Can get current running mode in PNLib.

2.3.13 PNLoadCalibrationData

Load calibration data.

```
PNLIB_API void PNLoadCalibrationData();
```

Remarks

If placements of sensors do not move, this function can be called to save the calibration data of current avatar, next time the sensor configuration can be loaded directly.

2.3.14 PNEnableClimbContact

Enable climb contact meanwhile disable ground contact.

```
PNLIB_API void PNEnableClimbContact(int avatarIndex, PNBOOL enable);
```

Parameters

avatarIndex

Avatar index.

enable

Whether enabling.

Remarks

If climb is needed when capturing motion, must enable this before.

2.3.15 PNResetClimbContact

Reset parameters about climb contact

```
PNLIB_API void PNResetClimbContact(int avatarIndex);
```

Parameters

avatarIndex

Avatar index need reset.

Remarks

PNLib will recalculate constraint parameters when calling this function.

2.3.16 PNEnableMagneticImmune

Enable magnetic immunity and set level.
If infected with surrounding ferromagnetic materials, posture accuracy of feet would be harmed. Then enable this are supposed to solve the problem.

```
PNLIB_API void PNEnableMagneticImmune(int avatarIndex,
                                      MagneticImmuneLevel level);
```

Parameters

avatarIndex

Avatar index of magnetic immunity need enable.

level

There are three magnetic immunity levels: disable, weak and strong.

Remarks

Default value is disabled. Weak level makes feet immunity and strong level makes both feet and legs.

2.3.17 PNSetSpineSmoothFactors

Set smooth factors of spine.

```
PNLIB_API void PNSetSpineSmoothFactors( int avatarIndex, float breastbone,
                                        float vertebra);
```

Parameters

avatarIndex

Avatar index.

breastbone

Smooth factor of Spine2 and Spine3, value range is 0~1.

vertebra

Smooth factor of Spine and Spine1, value range is 0~1.

2.3.18 PNSetSensorAcceleratorType

Accelerator type of sensor bound to bone, modify it if using different sensor.

```
PNLIB_API void PNSetSensorAcceleratorType(int boneId,
                                          SensorAcceleratorTypes type);
```

Parameters

sensorId
Sensor's id.
type
Accelerator type of sensor.

Remarks

Default accelerator type of bones is SA_Range8G. Set different range of sensors if needed.

2.3.19 PNGetSensorAcceleratorType

Get accelerator type of sensor.

```
PNLIB_API SensorAcceleratorTypes PNGetSensorAcceleratorType
                                                    (int boneId);
```

Parameters

sensorId
Sensor's id.

Return Value

This function returns current accelerator type of sensor.

Remarks

Default value is SA_Legacy8G.

2.3.20 PNReleaseScene

Release the resource of PNLib.

```
PNLIB_API void PNReleaseScene();
```

Remarks

Calling this function to release resources when closing the PNLib.

```
/*********************************************************
 *       Register and config data output callbacks       *
*********************************************************/
```

2.3.21 PNRegisterCalibrationProgressHandle

Register calibration progress callback.

```
PNLIB_API void PNRegisterCalibrationProgressHandle(void* customObject,
                            PNEventCalibrationProgressCallback handle);
```

Parameters

customObject
User can defined any type pointer needed.
handle
A function pointer of PNEventCalibrationProgressCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in calibration progress.

Example

```
void _stdcall _CalibrationProgressCallback(void* customObject, int avatarIndex,
                                           double percent)
{
    ControlDlg* pControlDlg = (ControlDlg*) customObject;
    TCHAR calibrationprogress[10];
    swprintf_s(calibrationprogress, 10, _T("%d\0"), percent);
    pControlDlg->m_static_calibrationprogress.SetWindowText(calibrationprogress);
}

BOOL ControlDlg::OnInitDialog()
{
    CDialogEx::OnInitDialog();
    PNRegisterCalibrationProgressHandle(this, _CalibrationProgressCallback);
    return TRUE;
}
```

2.3.22 PNRegisterBvhStringDataBoardcastHandle

Register BVH string data callback handle if need deal with string type.

```
PNLIB_API void PNRegisterBvhStringDataBoardcastHandle
     (void* customObject, PNEventBVHStringDataBoardcastCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventBVHStringDataBoardcastCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically after PNLib calculate to BVH data.

Example

```
void __stdcall _BVHStringDataBoardcast(void* customObject, int avatarIndex,
                                       char* bvhData)
{
    printf("Start output no. %d avatar's BVH data:\n%s\n", avatarIndex, bvhData);
}
```

```
int _tmain(int argc, _TCHAR* argv[])
{
    PNLibInit();
    PNSetRunningMode(RM_RawPlaying);
    PNEnableBvhDataBoardcast(TRUE);
    PNSetBvhDataBlockBoardcastType(BO_StringType);
    PNRegisterBvhStringDataBoardcastHandle(NULL, _BVHStringDataBoardcast);
    int avatarCount = PNOpenRawDataFile("RawData.raw");
    PNRawDataPlayStart();
    Sleep(40000);
    PNLibRelease();
    return 0;
}
```

2.3.23 PNRegisterBvhBinaryDataBoardcastHandle

Register BVH binary data callback handle if need to deal with binary type.

```
PNLIB_API void PNRegisterBvhBinaryDataBoardcastHandle(void* customObject,
                            PNEventBVHBinaryDataBoardcastCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventBVHBinaryDataBoardcastCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically after PNLib calculate to BVH data.

Example

```
void __stdcall _BVHBinaryDataBoardcast(void* customObject, int avatarIndex,
                                struct BvhOutputBinaryHeader* bbp, int packLen)
{
    printf("Start output no. %d avatar’s BVH data:\n%s\n", avatarIndex, bvhData);
}

int _tmain(int argc, _TCHAR* argv[])
{
    PNLibInit();
    PNSetRunningMode(RM_RawPlaying);
    PNEnableBvhDataBoardcast(TRUE);
    PNSetBvhDataBlockBoardcastType(BO_BinaryType);
```

```
    PNRegisterBvhBinaryDataBoardcastHandle (NULL, _BVHBinaryDataBoardcast);
    int avatarCount = PNOpenRawDataFile("RawData.raw");
    PNRawDataPlayStart();
    Sleep(40000);
    PNLibRelease();
    return 0;
}
```

2.3.24 PNRegisterBvhMatrixDataBoardcastHandle

Register matrix type of BVH data callback handle if need get matrix string data.

```
PNLIB_API void PNRegisterBvhMatrixDataBoardcastHandle(void* customObject,
                            PNEventBVHMatrixDataBoardcastCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventBVHMatrixDataBoardcastCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically after PNLib calculate to BVH data.

Example

```
void __stdcall _BVHMatrixDataBoardcast(void* customObject, int avatarIndex,
                                       char* matrixData)
{
    printf("Start output no. %d avatar’s Matrix data:\n%s\n", avatarIndex,
matrixData);
}

int _tmain(int argc, _TCHAR* argv[])
{
    PNLibInit();
    PNSetRunningMode(RM_RawPlaying);
    PNEnableBvhDataBoardcast(TRUE);
    PNSetBvhDataBlockBoardcastType(BO_MatrixStringType);
    PNRegisterBvhMatrixDataBoardcastHandle(NULL, _BVHMatrixDataBoardcast);
    int avatarCount = PNOpenRawDataFile("RawData.raw");
    PNRawDataPlayStart();
    Sleep(40000);
    PNLibRelease();
```

```
    return 0;
}
```

2.3.25 PNSetBvhDataBlockBoardcastType

Change output type of BVH data. Must register the relevant callback handle before change to a certain one.

```
PNLIB_API void PNSetBvhDataBlockBoardcastType(BvhDataStreamTypes type);
```

Parameters

type

Data types of BVH output stream: Binary, String and MatrixString.

Remarks

Must specify the corresponding type if registering one type of bvh broadcast callback function.

Example

```
PNEnableBvhDataBoardcast(TRUE);
PNSetBvhDataBlockBoardcastType(BO_MatrixStringType);
PNRegisterBvhMatrixDataBoardcastHandle(NULL, _BVHMatrixDataBoardcast);
```

2.3.26 PNEnableBvhDataBoardcast

Disable BVH callback if it is not needed as it occupies lots of CPU.

```
PNLIB_API void PNEnableBvhDataBoardcast(PNBOOL isEnable);
```

Parameters

isEnable

Whether enable the bvh broadcast.

Remarks

Must enable it if need bvh broadcast callback function.

Example

```
PNEnableBvhDataBoardcast(TRUE);
PNSetBvhDataBlockBoardcastType(BO_MatrixStringType);
PNRegisterBvhMatrixDataBoardcastHandle(NULL, _BVHMatrixDataBoardcast);
```

2.3.27 PNSetCalculatedDataBlockBoardcastType

Change output type of Calculated data.

```
PNLIB_API void PNSetCalculatedDataBlockBoardcastType
                                    (CalculatedDataStreamTypes type);
```

Parameters

type

Data types of Calculated output stream: Binary and String.

Remarks

Must specify the corresponding type if registering one type of calculated broadcast callback function.

2.3.28 PNRegisterCalculatedStringDataBoardcastHandle

Register calculation data callback handle.

```
PNLIB_API void PNRegisterCalculatedStringDataBoardcastHandle
      (void* customObject, PNEventCalculatedStringDataCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventCalculatedStringDataCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically after PNLib calculate to calculated data.

2.3.29 PNRegisterCalculatedBinaryDataBoardcastHandle

Register calculation data callback handle.

```
PNLIB_API void PNRegisterCalculatedBinaryDataBoardcastHandle
      (void* customObject, PNEventCalculatedBinaryDataCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventCalculatedBinaryDataCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically after PNLib calculate to calculated data.

2.3.30 PNSetCalculatedQuaternionDataType

Change quaternion type in calculation frame data.

```
PNLIB_API void PNSetCalculatedQuaternionDataType
                                        (OutputQuaternionTypes type);
```

Parameters

type

Quaternion types of output stream: Global Raw Quaternion, Global Bone Quaternion and Local Bone Quaternion.

2.3.31 PNSetCalculatedAccelerationDataType

Change Acceleration type in calculation frame data.

```
PNLIB_API void PNSetCalculatedAccelerationDataType
                                        (OutputAccelerationTypes type);
```

Parameters

type

Acceleration data types of output stream: Module Raw Data and Global Data.

2.3.32 PNSetCalculatedGyroDataType

Change Gyro type in calculation frame data.

```
PNLIB_API void PNSetCalculatedGyroDataType(OutputGyroType type);
```

Parameters

type

Gyro data types of output stream: Module Raw Data and Global Data.

2.3.33 PNEnableCalculationDataBoardcast

Enable/disable calculation data callback.

```
PNLIB_API void PNEnableCalculationDataBoardcast(PNBOOL enable);
```

Parameters

enablbe

Whether enable the calculation data broadcast.

Remarks

Must enable it if calculation data broadcast callback function is needed.

2.3.34 PNRegisterRawDataPlayingProgressHandle

Register raw data playing progress callback.

```
PNLIB_API void PNRegisterRawDataPlayingProgressHandle
              (void* customObject, PNEventPlayProgressCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventPlayProgressCallback type.

Remarks

Register this function when program initializes. The callback

function defined with the prescribed format will be called automatically in raw data playing progress in playing thread.

Example

```
void _stdcall _PlayProgressCallback(void* customObject, int currentFrame, int
                                    totalFrames)
{
    ControlDlg* pControlDlg = (ControlDlg*) customObject;
    If(currentFrame == totalFrames)
    pControlDlg->m_static_finish.SetWindowText("Endplay");
}

BOOL ControlDlg::OnInitDialog()
{
    CDialogEx::OnInitDialog();
    PNRegisterRawDataPlayingProgressHandle(this, _PlayProgressCallback);
    return TRUE;
}
```

2.3.35 PNRegisterPlayingRawDataParsedHandle

Register raw data parsed data callback.

```
PNLIB_API void PNRegisterPlayingRawDataParsedHandle(void* customObject,
                                    PNEventRawDataParsedCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventRawDataParsedCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in raw data parsed progress.

2.3.36 PNRegisterContactEditCallback

Register contact edit callback.

```
PNLIB_API void PNRegisterContactEditCallback(void* customObject,
                            PNEventConstraintDataCallback editHandler);
```

Parameters

customObject

User can defined any type pointer needed.

editHandler

A function pointer of PNEventConstraintDataCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in contact edit.

Example

```
void __stdcall _PNConstraintDataCallback(void* customObject, int avatarIndex,
                                         FrameContactData* contactData)
{
    for(int i = 0; i < 5; i++)
    printf("no. %d avatar's %d frame's %d point: contact: %d; edited %d\n",
avatarIndex, ContactData->FrameIndex, i, ContactData->ContactInfo[i].IsContact,
ContactData->ContactInfo[i].IsEdited);
}

int _tmain(int argc, _TCHAR* argv[])
{
    PNLibInit();
    PNSetRunningMode(RM_RawPlaying);
    int avatarCount = PNOpenRawDataFile("RawData.raw");
    PNRegisterContactEditCallback(NULL, _PNConstraintDataCallback);
    editstate = PNEditContact(1, 0, CP_Hip, TRUE);
    PNCloseRawDataFile();
    PNLibRelease();
    return 0;
}
```

2.3.37 PNRegisterActionRecognizationStringDataBoardcastHandle

Register action recognition event callback.

```
PNLIB_API void PNRegisterActionRecognizationStringDataBoardcastHandle
            (void* customObject,
            PNEventActionRecognizationDataStringStreamCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventActionRecognizationDataStringStreamCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in action recognization event.

Example

```
void __stdcall _ActionRecognitionDataStringStreamCallback (void* customObject,
                                                    int avatarIndex, char* actionData)
{
    printf("Start output no. %d avatar’s action recognition data:\n%s\n",
avatarIndex, actionData);
}

int _tmain(int argc, _TCHAR* argv[])
{
    PNLibInit();
    PNSetRunningMode(RM_RawPlaying);
    PNEnableActionRecognition(TRUE);
    PNRegisterActionRecognitionStringDataBoardcastHandle(NULL,
    _ActionRecognitionDataStringStreamCallback);
    int avatarCount = PNOpenRawDataFile("RawData.raw");
    PNRawDataPlayStart();
    Sleep(40000);
    PNLibRelease();
    return 0;
}
```

2.3.38 PNRegisterBodyMassVectorStringCallback

Register body mass vector of action recognition callback.

```
PNLIB_API void PNRegisterBodyMassVectorStringCallback
      (void* customObject, PNEventBodyMassVectorStringCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventBodyMassVectorStringCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in action recognization event if body mass vector is needed.

2.3.39 PNRegisterBodySwingVectorStringCallback

Register body swing vector of action recognition callback.

```
PNLIB_API void PNRegisterBodySwingVectorStringCallback
     (void* customObject, PNEventBodySwingVectorStringCallback handle);
```

Parameters

customObject

User can defined any type pointer needed.

handle

A function pointer of PNEventBodySwingVectorStringCallback type.

Remarks

Register this function when program initializes. The callback function defined with the prescribed format will be called automatically in action recognization event if the body swing vector is needed.

```
/*********************************************************
 *                  Manage avatars                       *
 *********************************************************/
```

2.3.40 PNCreateAvatar

Create an avatar in scene.

```
PNLIB_API int PNCreateAvatar();
```

Return Value

Return current avatar index.

Remarks

Cannot see this avatar in screen if it has no motion data.

2.3.41 PNRemoveAvatar

Delete an avatar from scene.

```
PNLIB_API int PNRemoveAvatar(int avatarIndex);
```

Parameters

avatarIndex

Avatar index need delete.

Return Value

Return remainder avatar count.

Remarks

It will not delete the avatar from data file unless saving file.

2.3.42 PNGetAvatarCount

Get total avatars count in current scene.

```
PNLIB_API int PNGetAvatarCount();
```

Return Value

Return current avatar count in PNLib.

Remarks

Include avatar that created.

2.3.43 PNSetAvatarName

Set a name for avatar.

```
PNLIB_API void PNSetAvatarName(int avatarIndex, char* name);
```

Parameters

avatarIndex

Avatar index.

name

Avatar name corresponding to the index.

2.3.44 PNGetAvatarName

Get avatar's name.

```
PNLIB_API void PNGetAvatarName(int avatarIndex, char* name,
                               int buffLen);
```

Parameters

avatarIndex

Avatar index.

name

Head pointer of buffer saving the name.

buffLen

Buffer capability.

Remarks

Name is an output parameter and length of which must be equal to or smaller than the actual name buffer, otherwise an incomplete name will be got.

2.3.45 PNSetBoneDimensions

Set avatar's bone dimensions.

```
PNLIB_API void PNSetBoneDimensions(int avatarIndex,
                                   BoneDimension* dimensions);
```

Parameters

avatarIndex

Avatar index.

dimensions

Pointer of BoneDimension type.

Remarks

Define a BoneDimension structure and input its pointer to PNLib.

2.3.46 PNGetBoneDimensions

Get avatar's bone dimensions.

```
PNLIB_API void PNGetBoneDimensions(int avatarIndex,
                                   BoneDimension* dimsBuffer);
```

Parameters

avatarIndex

Avatar index.

dimsBuff

Pointer of BoneDimension type used to save bone dimensions from PNLib.

Remarks

A default bone dimension will be got if creating an avatar and not set its bone dimensions.

2.3.47 PNGetBoneLength

Get avatar's bone length.

```
PNLIB_API float PNGetBoneLength(int avatarIndex, int boneIndex);
```

Parameters

avatarIndex

Avatar index.

boneIndex

Bone index in Standard Bone Table.

Return Value

Return the length of the bone corresponding to the bone index.

Remarks

59 bones' length can be got in Neuron with fingers. 21 bones' length in Legacy without fingers. Call this function to get their lengths.

Fingers will be scaled by palm length. Others are calculated by the BoneDimension structure as below:

Bone Index	Bone Name	Calculation
0	Hips	Body * 1/5
1	RightUpLeg	UpperLeg
2	RightLeg	LowerLeg
3	RightFoot	FootLength
4	LeftUpLeg	UpperLeg
5	LeftLeg	LowerLeg
6	LeftFoot	FootLength
7	RightShoulder	ShoulderWidth * 1/2
8	RightArm	UpperArm

9	RightForeArm	Forearm
10	RightHand	Palm
11	LeftShoulder	ShoulderWidth * 1/2
12	LeftArm	UpperArm
13	LeftForeArm	Forearm
14	LeftHand	Palm
15	Head	Head
16	Neck	Neck
17	Spine3	Body * 1/5
18	Spine2	Body * 1/5
19	Spine1	Body * 1/5
20	Spine	Body * 1/5

2.3.48 PNBindSensor

Binding a sensor to bone. Return FALSE if failed.

```
PNLIB_API PNBOOL PNBindSensor(int avatarIndex, int boneIndex,
                              int sensorId);
```

Return Value

Return TRUE if binding succeeds, otherwise return FALSE.

Parameters

avatarIndex

Avatar index.

boneIndex

Bone index.

sensorId

Sensor id.

Remarks

Ensure right avatarIndex, boneIndex and sensorId or the data accuracy might be harmed.

2.3.49 PNRemoveSensor

Remove sensor from referred bone. Return FALSE if failed.

```
PNLIB_API PNBOOL PNRemoveSensor(int avatarIndex, int boneIndex);
```

Return Value

Return TRUE if removing succeeds, otherwise return FALSE.

Parameters

avatarIndex

Avatar index, start from 0.

boneIndex

Bone index, start from 0.

Remarks

The sensorId of this boneIndex will be 0 after calling this function.

2.3.50 PNIsBindingSensor

Check the referred bone if is bound sensor.

```
PNLIB_API PNBOOL PNIsBindingSensor(int avatarIndex, int boneIndex);
```

Return Value

Return TRUE if binding currently, otherwise return FALSE.

Parameters

avatarIndex

Avatar index, start from 0.

boneIndex

Bone index, start from 0.

2.3.51 PNCheckSensorBindingMode

Check the current sensor binding if is suitable with the referred sensor combination mode.

```
PNLIB_API PNBOOL PNCheckSensorBindingMode(int avatarIndex,
                                          SensorCombinationModes mode);
```

Return Value

Return TRUE if suitable, otherwise return FALSE.

Parameters

avatarIndex

Avatar index, start from 0.

mode

Sensor combination mode.

Remarks

For SC_ArmOnly mode, 2 nodes of left upper arm and left forearm are necessary, or, right upper arm and right forearm are necessary.

For SC_UpperBody mode, 4 nodes of chest, hips, left upper arm and left forearm are necessary, or, chest, hips, right upper arm and right forearm are necessary.

For SC_FullBody mode, 6 nodes of left upper leg, left leg, right upper leg, right leg, hips and chest sensor are necessary.

2.3.52 PNResetBoneMapping

Reset sensors binding to default bone map. Return FALSE if failed.

```
PNLIB_API PNBOOL PNResetBoneMapping(int avatarIndex);
```

Return Value

Return TRUE if mapping succeeds, otherwise return FALSE.

Parameters

avatarIndex

Avatar index.

Remarks

All sensorId of this avatar's bone will be 0 after calling this function.

2.3.53 PNGetBoneName

Get bone name by avatar index and bone index.

```
PNLIB_API char* PNGetBoneName(int avatarIndex, int boneIndex);
```

Return Value

Get bone name corresponding with a bone of avatar.

Parameters

avatarIndex

Avatar index.

boneIndex

Bone index.

Remarks

All bone name refer to Standard Bone Table.

2.3.54 PNGetBoneNameBySensorId

Get bone name by avatar index and sensor id.

```
PNLIB_API char* PNGetBoneNameBySensorId(int avatarIndex, int sensorId);
```

Return Value

Get bone name corresponding with a bound sensor id of avatar.

Parameters

avatarIndex

Avatar index.

sensorId

Sensor id starting from 1.

Example

```
PNBOOL bindresult = PNBindSensor(avatarIndex, boneindex, sensorid);
Char* actualname = PNGetBoneNameBySensorId(avatarIndex, sensorid);
```

2.3.55 PNGetSensorId

Get sensor id by avatar index and bone index.

```
PNLIB_API int PNGetSensorId(int avatarIndex, int boneIndex);
```

Return Value

Get bound sensor id corresponding with bone index of avatar.

Parameters

avatarIndex

Avatar index.

boneIndex

Bone index starting from 0.

Remarks

Get sensor ID based on bone index, starting from 1. Return 0 means no sensor bound on the referred bone, -1 means error occurred.

2.3.56 PNGetBoneIndexBySensorId

Get bone index by avatar index and sensor id.

```
PNLIB_API int PNGetBoneIndexBySensorId(int avatarIndex, int sensorId);
```

Return Value

Bone index.

Parameters

avatarIndex

Avatar index.

sensorId

Sensor id.

Remarks

Return -1 if this sensor was not bound to a bone.

2.3.57 PNGetHipWidth

Get hips width of avatar.

```
PNLIB_API float PNGetHipWidth(int avatarIndex);
```

Return Value

Return a float type variable indicating hips width.

Parameters

avatarIndex

Avatar index.

Remarks

Return 0 if invalid avatar index. The default value is 0.23f. Set it with PNSetBoneDimensions.

2.3.58 PNGetHipHeight

Get hips height of avatar, the height is the distance from hips to ground.

```
PNLIB_API float PNGetHipHeight(int avatarIndex);
```

Return Value

Return a float type variable indicating the length of hips from ground.

Parameters

avatarIndex

Avatar index.

Remarks

Return 0 if invalid avatar index. The default value is 0.96f which is the length of UpperLeg plus LowerLeg. Set it with PNSetBoneDimensions.

2.3.59 PNGetShoulderWidth

Get shoulder width of avatar.

```
PNLIB_API float PNGetShoulderWidth(int avatarIndex);
```

Return Value

Return a float type variable indicating the shoulder width.

Parameters

avatarIndex

Avatar index.

Remarks

Return 0 if invalid avatar index. The default value is 0.35f. Set it with PNSetBoneDimensions.

2.3.60 PNGetHeelHeight

Get heel height of avatar.

```
PNLIB_API float PNGetHeelHeight(int avatarIndex);
```

Return Value

Return a float type variable indicating the heel height.

Parameters

avatarIndex

Avatar index.

Remarks

Return 0 if invalid avatar index. The default value is 0.05f. Set it with PNSetBoneDimensions.

2.3.61 PNGetInitiationDirection

Get direction of avatar at time of calibration.

```
PNLIB_API void PNGetInitiationDirection(int avatarIndex,
                                        Vector3_t* zd);
```

Parameters

avatarIndex

Avatar index.

zd

Vector3 type pointer of direction.

2.3.62 PNGetInitiationLeftDirection

Get the left direction of avatar at time of calibration.

```
PNLIB_API void PNGetInitiationLeftDirection(int avatarIndex,
                                            Vector3_t* xd);
```

Parameters

avatarIndex

Avatar index.

xd

Vector3 type pointer of left direction.

2.3.63 PNCanCalibratePose

Check whether a pose of calibration is necessary based on current sensor bound.

```
PNLIB_API PNBOOL PNCanCalibratePose(int avatarIndex,
                                    CalibrationTypes type);
```

Return Value

Return TRUE if this calibration is necessary, otherwise return FALSE.

Parameters

avatarIndex

Avatar index.

type

Calibration Type.

Remarks

If left upper arm and left forearm or right upper arm and right fore arm are bound, A pose and T pose are necessary.

If left upper leg, left leg, right upper leg, right leg, hips and spine sensor are bound, S pose is necessary.

2.3.64 PNGetCalibrationData

Get calibration data.

```
PNLIB_API void PNGetCalibrationData(int avatarIndex,
                                    struct CalibrationData* data);
```

Parameters

avatarIndex
Avatar index.
data
Pointer of CalibrationData struct data.

2.3.65 PNSetCalibrationData

Set calibration data.

```
PNLIB_API void PNSetCalibrationData(int avatarIndex,
                                    struct CalibrationData* data);
```

Parameters

avatarIndex
Avatar index.
data
Pointer of CalibrationData struct data.

2.3.66 PNClearIntegralState

Clear integral state.

```
PNLIB_API void PNClearIntegralState(int avatarIndex);
```

Parameters

avatarIndex
Avatar index.

Remarks

An error code and message will be got if invalid avatar index is sent in.

2.3.67 PNSetDataOutputFrequencyRatio

Set data output frequency ratio.

```
PNLIB_API void PNSetDataOutputFrequencyRatio(int ratio);
```

Parameters

ratio
Ratio of data output frequency.

Remarks

Zero or negative values will be set failure and send error code. The actual output frequency is the set frequency with PNSetDataAcquisitionFrequency dividing by ratio, such as:

frequency	ratio	formula	result
30	1	30/1	30
48	2	48/2	24
60	4	60/4	15
96	8	96/8	12

2.3.68 PNGetDataOutputFrequencyRatio

Get data output frequency ratio.

```
PNLIB_API int PNGetDataOutputFrequencyRatio();
```

Return Value

Return ratio of data output frequency.

Remarks

Default value is 1 after PNLib initialize.

```
/*******************************************************
 *           Calculation parameters settings           *
 *******************************************************/
```

2.3.69 PNGetKalmanParams

Get kalman parameters.

```
PNLIB_API struct KalmanParams* PNGetKalmanParams(int avatarIndex);
```

Return Value

Pointer of KalmanParams struct.

Parameters

avatarIndex

Avatar index.

2.3.70 PNSetKalmanParams

Set kalman parameters.

```
PNLIB_API void PNSetKalmanParams(int avatarIndex,
                                 struct KalmanParams params);
```

Parameters

avatarIndex

Avatar index.

params

KalmanParams struct.

2.3.71 PNResetKalmanParams

Reset kalman parameters.

```
PNLIB_API void PNResetKalmanParams(int avatarIndex);
```

Parameters

avatarIndex

Avatar index.

2.3.72 PNSetPDamping

Set damping coefficient of preventing P diffuse.

```
PNLIB_API void PNSetPDamping(int avatarIndex, float pdamping);
```

Parameters

avatarIndex

Avatar index.

pdamping

Float type value.

2.3.73 PNGetPDamping

Get damping coefficient of preventing P diffuse.

```
PNLIB_API float PNGetPDamping(int avatarIndex);
```

Return Value

Return current damping coefficient.

Parameters

avatarIndex

Avatar index.

2.3.74 PNSetJointStiffness

Set joint stiffness.

```
PNLIB_API void PNSetJointStiffness(int avatarIndex, float percent);
```

Parameters

avatarIndex

Avatar index.

percent

Percent of joint stiffness.

2.3.75 PNSetStepStiffness

Set step stiffness.

```
PNLIB_API void PNSetStepStiffness(int avatarIndex, float percent);
```

Parameters

avatarIndex

Avatar index.

percent

Percent of step stiffness.

2.3.76 PNSetStepConstraint

Set step constraint.

```
PNLIB_API void PNSetStepConstraint(int avatarIndex, float percent);
```

Parameters

avatarIndex

Avatar index.

percent

Percent of step constraint.

2.3.77 PNGetStiffnessPercent

Get stiffness percent.

```
PNLIB_API void PNGetStiffnessPercent(int avatarIndex,
                                     float* jointStiffnessPercent,
                                     float* stepStiffnessPercent,
                                     float* stepConstraintPercent);
```

Parameters

avatarIndex

Avatar index.

jointStiffnessPercent

Float type pointer to save joint stiffness percent.

stepStiffnessPercent

Float type pointer to save step stiffness percent.

stepConstraintPercent

Float type pointer to save step constraint percent.

```
/*********************************************************
 *                 Smooth settings                       *
 *********************************************************/
```

2.3.78 PNEnableSmoothFilter

Enable smooth filter.

```
PNLIB_API void PNEnableSmoothFilter(int avatarIndex, PNBOOL enable);
```

Parameters

avatarIndex

Avatar index.

enable

BOOL type value. Set TRUE if enable smooth filter, otherwise set FALSE. It is disabled by default.

2.3.79 PNIsSmoothFilterEnabled

Check whether smooth filter is enabled.

```
PNLIB_API PNBOOL PNIsSmoothFilterEnabled(int avatarIndex);
```

Return Value

Return TRUE if smooth filter is enabled, otherwise return FALSE.

Parameters

avatarIndex

Avatar index.

2.3.80 PNSetSmoothFactor

Set smooth factor.

```
PNLIB_API void PNSetSmoothFactor(int avatarIndex,
                                 SmoothFactors smoothFactors);
```

Parameters

avatarIndex

Avatar index.

smoothFactors

SmoothFactors struct.

Remarks

Need set smooth factor if smooth filter is enabled.

2.3.81 PNGetSmoothFactor

Get current smooth factor in PNLib.

```
PNLIB_API void PNGetSmoothFactor(int avatarIndex,
                                 SmoothFactors* smoothFactors);
```

Parameters

avatarIndex

Avatar index.

smoothFactors

Pointer of SmoothFactors struct.

2.3.82 PNEnableFootLock

Enable/disable foot lock while certain foot contacts ground.

```
PNLIB_API void PNEnableFootLock(PNBOOL enable);
```

Parameters

enable

BOOL type value. Set TRUE if enable foot lock, otherwise set FALSE. It is enabled by default.

```
/*******************************************************
 *               Manage drawing scene                  *
```

```
**********************************************************/
```

2.3.83 PNCreateBvhPlayer

Create bvh player.

```
PNLIB_API void* PNCreateBvhPlayer(BVHWindowContainerRef hWndParent);
```

Return Value

Window handle of bvhplayer. Null if creation fails.

Parameters

hWndParent

HWND of BVH Window to show in.

Remarks

Create a bvh player in memory with this function. Call PNBvhPlayerResizeToParent if need to show it in window.

2.3.84 PNSetBvhPlayerCameras

Split and duplicate drawing scene.

```
PNLIB_API void PNSetBvhPlayerCameras(int cameras);
```

Parameters

cameras

Number of camera, most is 4.

2.3.85 PNEnableBvhPlayerCameraBind

Bind/unbind camera to an avatar.

```
PNLIB_API void PNEnableBvhPlayerCameraBind(int cameraIndex,
                                           int avatarIndex,
                                           PNBOOL enableBind);
```

Parameters

cameraIndex

Camera index, start from 0.

avatarIndex

Avatar index.

enableBind

Set TRUE to bind or set FALSE to unbind.

2.3.86 PNBvhPlayerResizeToParent

Resize drawing window to fill the container.

```
PNLIB_API void PNBvhPlayerResizeToParent();
```

Remarks

To show bvh player in window, must call PNCreateBvhPlayer firstly.

2.3.87 PNCloseBvhPlayer

Release drawing scene.

```
PNLIB_API void PNCloseBvhPlayer();
```

Remarks

Call this function to close bvh player.

2.3.88 PNEnableMassShowing

Enable/disable mass showing for some avatar.

```
PNLIB_API void PNEnableMassShowing(int avatarIndex, PNBOOL enable);
```

Parameters

avatarIndex

Avatar index.

enable

Set TRUE if showing mass otherwise set FALSE.

2.3.89 PNEnableRendering

Enable/disable rendering.

```
PNLIB_API void PNEnableRendering(PNBOOL enable);
```

Parameters

enable

Set TRUE if rendering otherwise set FALSE.

```
/*********************************************************
 *                 Realtime data interface               *
 *********************************************************/
```

2.3.90 PNPushData

Push raw data into PNLib.

```
PNLIB_API void PNPushData(unsigned char* data);
```

Parameters

data

Pointer of one data package.

Remarks

Data entrance, if avatar number is unknown, push data that received from serial port or network to PNLib in a single package method.

2.3.91 PNPushDataForAvatar

Push avatar's raw data into PNLib.

```
PNLIB_API void PNPushDataForAvatar(int avatarIndex,
                                   unsigned char* data);
```

Parameters

avatarIndex

Avatar index.

data

Pointer of single data package for corresponding avatar.

Remarks

If corresponding avatar index is known, push each data to PNLib with this function.

2.3.92 PNEnableLostDataFitting

Enable/disable fitting of lost data.

```
PNLIB_API void PNEnableLostDataFitting(PNBOOL enable);
```

Parameters

enable

Set TRUE to enable lost data fitting otherwise set FALSE.

2.3.93 PNClearCalibrationBufferedData

Clear the buffered data used to calibration.

```
PNLIB_API void PNResetCalibrationSteps(int avatarIndex);
```

Remarks

Prior to a new calibration, it is required to clear the previous buffered data by this function.

2.3.94 PNCalibrateAvatar

Start a calibrating action of an avatar.

```
PNLIB_API void PNCalibrateAvatar(int avatarIndex, CalibrationTypes type);
```

Parameters

avatarIndex

Avatar index.

type

Calibration type.

2.3.95 PNCalibrateAllAvatars

Start a calibrating action for all avatar.

```
PNLIB_API void PNCalibrateAllAvatars(CalibrationTypes type);
```

Parameters

type

Calibration type.

2.3.96 PNGetSensorReceivingStatus

Get receiving percentage of sensor data.

```
PNLIB_API int PNGetSensorReceivingStatus(int avatarIndex,
                  float bufferForPercentData [FULL_BODY_BONE_COUNT]);
```

Return Value

Return current sensor number.

Parameters

avatarIndex

Avatar index.

bufferForPecentData

Float type array buffer for percent data.

Remarks

FULL_BODY_BONE_COUNT is 167 defined in PNDataTypes.h.

```
/*********************************************************
 *               Raw file interface              *
 *********************************************************/
```

2.3.97 PNGetRawFileInfo

Get the referred raw file information. Such as record time, sensor type, freq, avatar count, etc.

```
PNLIB_API RawFileInfo* PNGetRawFileInfo(char* filename);
```

Return Value

Return a pointer of RawFileInfo type.

Parameters

Filename

File path.

2.3.98 PNOpenRawDataFile

Open raw file to replay the captured motion data, return the avatar number in this motion data file.

```
PNLIB_API int PNOpenRawDataFile(char* filename);
```

Return Value

Return avatar number of current file if open successfully, and return 0 if failed.

Parameters

filename
File path.

Example

```
PNLibInit();
PNSetRunningMode(RM_RawPlaying);
int avatarCount = PNOpenRawDataFile("RawDataFile.raw");
```

2.3.99 PNRawDataPlayGetTotalFrames

Get total frames in opened raw file.

```
PNLIB_API unsigned long PNRawDataPlayGetTotalFrames();
```

Return Value

Return total frames of current file.

2.3.100 PNGetSensorSuitType

Get sensor suit type in opened raw file.

```
PNLIB_API SensorSuitTypes PNGetSensorSuitType();
```

Return Value

Return current sensor suit type of file.

2.3.101 PNGetSensorCombinationMode

Get node combination mode.

```
PNLIB_API SensorCombinationModes PNGetSensorCombinationMode
                                                    (int avatarIndex);
```

Return Value

Return sensor combination mode of corresponding avatar.

Parameters

avatarIndex
Avatar index.

Remarks

Default value is SC_Unknown.

2.3.102 PNRawDataPlaySetPlayingPosition

Set playing position.

```
PNLIB_API BOOL PNRawDataPlaySetPlayingPosition (int pos);
```

Return Value

Return TRUE if set successfully, otherwise return FALSE.

Parameters

pos

Frame index to play. Start from 0. Biggest number is totalFrames subtracting 1.

2.3.103 PNRawDataPlayGetCurrentPlayingPosition

Get current playing position.

```
PNLIB_API unsigned long PNRawDataPlayGetPlayingPosition();
```

Return Value

Return current playing frame index which start from 0.

2.3.104 PNRawDataPlaySetSpeed

Set playing speed ratio.

```
PNLIB_API void PNRawDataPlaySetSpeed(float ratio);
```

Parameters

ratio

Current playing speed ratio. The more the faster.

2.3.105 PNRawDataPlayStart

Start playing data.

```
PNLIB_API void PNRawDataPlayStart();
```

Remarks

This function will return at once, but play raw file in another thread.

2.3.106 PNRawDataPlayPause

Raw data playing pause.

```
PNLIB_API void PNRawDataPlayPause();
```

2.3.107 PNRawDataPlayStop

Raw data playing stop.

```
PNLIB_API void PNRawDataPlayStop();
```

Remarks

This function will set playing position to frame 0 and call callback function.

2.3.108 PNRawDataPlayEnableReversePlaying

Enable reverse playing.

```
PNLIB_API void PNRawDataPlayEnableReversePlaying(BOOL enable);
```

Parameters

enable

A BOOL type variable. Set TRUE to enable it, otherwise set

FALSE.

2.3.109 PNRawDataPlaySetToPrev

Set to previous frame.

```
PNLIB_API unsigned long PNRawDataPlaySetToPrev();
```

Return Value

Return previous frame index.

2.3.110 PNRawDataPlaySetToNext

Set to next frame.

```
PNLIB_API unsigned long PNRawDataPlaySetToNext();
```

Return Value

Return next frame index.

2.3.111 PNCloseRawDataFile

Close raw data file and lean space of raw file.

```
PNLIB_API void PNCloseRawDataFile();
```

Remarks

Close current file before open another.

```
/*********************************************************
 *                Constraint editing                     *
 *********************************************************/
```

2.3.112 PNEditContact

Edit constraint contact.

```
PNLIB_API PNSTATUS PNEditContact(int avatarIndex, int frameIndex,
                                 ConstraintPoint point, PNBOOL isContact);
```

Return Value

Return error code if failed. Get last error message with PNGetLastErrorMessage.

Parameters

avatarIndex

Avatar index.

frameIndex

Frame index.

point

Enumeration variable of ConstraintPoint type.

isContact

BOOL type variable means whether contacting.

2.3.113 PNGetContactStatus

Get constraint contact status.

```
PNLIB_API PNSTATUS PNGetContactStatus (int avatarIndex, int frameIndex,
                                ConstraintPoint point, BOOL* isContact);
```

Return Value

Return error code. Return 0 if success, otherwise return corresponding error code.

Parameters

avatarIndex

Avatar index.

frameIndex

Frame index.

point

Enumeration variable of ConstraintPoint type.

isContact

Pointer of BOOL type send constraint status corresponding constraint point for avatar.

2.3.114 PNBatchEditContact

Edit a batch of constraint contact.

```
PNLIB_API PNSTATUS PNBatchEditContact(int avatarIndex,
                                int startFrameIndex, int endFrameIndex,
                                ConstraintPoint point, BOOL isContact);
```

Return Value

Return error code. Return 0 if success, otherwise return corresponding error code.

Parameters

avatarIndex

Avatar index.

startFrameIndex

Start frame index editing.

endFrameIndex

End frame index of editing frame.

point

Enumeration variable of ConstraintPoint type.

isContact

BOOL type variable means contacting status.

2.3.115 PNBatchResetContactEditStatus

```
PNLIB_API PNSTATUS PNBatchResetContactEditStatus (int avatarIndex,
        int startFrameIndex, int endFrameIndex, ConstraintPoint point);
```

Return Value

Return error code. Return 0 if success, otherwise return corresponding error code.

Parameters

avatarIndex

Avatar index.

startFrameIndex

Start frame index editing.

endFrameIndex

End frame index of editing frame.

point

Enumeration variable of ConstraintPoint type.

2.3.116 PNResetContactEditStatus

Cancel contact edit of some contact point of some frame.

```
PNLIB_API PNSTATUS PNResetContactEditStatus(int avatarIndex,
                                int frameIndex, ConstraintPoint point);
```

Return Value

Return error code. Return 0 if success, otherwise return corresponding error code.

Parameters

avatarIndex

Avatar index.

frameIndex

Frame index.

point

Enumeration variable of ConstraintPoint type.

2.3.117 PNFeetConstraintOptimization

Optimization feet constraint with the specified level.

```
PNLIB_API void PNFeetConstraintOptimization(int avatarIndex,
                                            unsigned int level);
```

Parameters

avatarIndex

Avatar index.

level

Unsigned int type. 0 means reset to no optimized status. 10 is

the max level can be used.

Remarks

Reset level will reset all the constraint status include result edited.

```
/*********************************************************
 *               Data export operations                  *
 *********************************************************/
```

2.3.118 PNExportRawData

Start to export raw data to file.

```
PNLIB_API char* PNExportRawData();
```

Return Value

Return a pointer of char type pointing file name saved.

2.3.119 PNStopExportRawData

Stop to export raw data to file.

```
PNLIB_API void PNStopExportRawData();
```

2.3.120 PNExportRawDataTxt

Start to export raw data about one sensor of one avatar.

```
PNLIB_API char* PNExportRawDataTxt(int avatarIndex, int sensorId);
```

Return Value

Return a pointer of char type pointing file name saved.

Remarks

Export a raw file about one sensor of one avatar.

2.3.121 PNStopExportRawDataTxt

Stop to export raw data about one sensor of one avatar.

```
PNLIB_API void PNStopExportRawDataTxt(int avatarIndex);
```

2.3.122 PNExportCalculationData

Start to export calculation data to file.

```
PNLIB_API char* PNExportCalculationData(int avatarIndex);
```

Return Value

Return a pointer of char type pointing file name saved.

Parameters

avatarIndex

Avatar index.

2.3.123 PNStopExportCalculationData

Stop to export calculation data to file.

```
PNLIB_API void PNStopExportCalculationData(int avatarIndex);
```

Parameters

avatarIndex

Avatar index.

2.3.124 PNExportBvhData

Start to export BVH data.

```
PNLIB_API char* PNExportBvhData(int avatarIndex);
```

Return Value

Return a pointer of char type pointing file name saved.

Parameters

avatarIndex

Avatar index.

2.3.125 PNStopExportBvhData

Stop to export BVH data.

```
PNLIB_API void  PNStopExportBvhData(int avatarIndex);
```

Return Value

Return a pointer of char type pointing file name saved.

Parameters

avatarIndex

Avatar index.

2.3.126 PNExportFbxData

Start to export fbx data.

```
PNLIB_API char* PNExportFbxData(int avatarIndex);
```

Return Value

Return a pointer of char type pointing file name saved.

Parameters

avatarIndex

Avatar index.

Remarks

Exporting to a Chinese or special symbol path will be failed.

2.3.127 PNStopExportFbxData

Stop to export fbx data.

```
PNLIB_API void PNStopExportFbxData(int avatarIndex);
```

Parameters

avatarIndex

Avatar index.

Remarks

Exporting to a Chinese or special symbol path will be failed.

2.3.128 PNSetBvhDataFormat

Set bvh data format.

```
PNLIB_API void PNSetBvhDataFormat(BOOL isWithDisp,
                                  enum RotateOrders order);
```

Parameters

isWithDisp

BOOL type variable means whether is with displacement. Set TRUE if is, otherwise set FALSE. Default with displacement if this function is not called.

order

Enum variable of RotateOrders type. Default value is YXZ if this function is not called.

2.3.129 PNSetBvhDataWithReference

With/without reference before export bvh data.

```
PNLIB_API void PNSetBvhDataWithReference(BOOL withReference);
```

Parameters

withReference

With prefixion or not. Set TRUE if is, otherwise set FALSE. Default with reference if this function is not called.

2.3.130 PNBvhBinaryDataOutputIsCompression

BVH data output compression flag.

```
PNLIB_API void PNBvhBinaryDataOutputIsCompression
                                            (PNBOOL isCompression);
```

Parameters

isCompression

Compress or not. Set TRUE if do, otherwise set FALSE.

2.3.131 PNEnableBvhDataGlobalDisplacement

Enable global or local coordinate of displacement that bvh data exporting.

```
PNLIB_API void PNEnableBvhDataGlobalDisplacement(int avatarIndex,
```

```
                                                    BOOL isGlobalDips);
```

Parameters

avatarIndex

Avatar index.

isGlobalDips

BOOL type variable. Set TRUE if is in global coordinate, otherwise set FALSE.

2.3.132 PNRotateFaceDirection

Set rotation of face direction.

```
PNLIB_API void PNRotateFaceDirection(int avatarIndex, float yaw);
```

Parameters

avatarIndex

Avatar index.

yaw

Float type variable means yawing degree of avatar.

2.3.133 PNRotateModel

Set rotation of avatar pitching and rolling degree.

```
PNLIB_API void PNRotateModel(int avatarIndex, float pitch,
                                                        float roll);
```

Parameters

avatarIndex

Avatar index.

pitch

Float type variable means pitching degree of avatar.

roll

Float type variable means rolling degree of avatar.

2.3.134 PNZeroOutAllAvatar

Zero out all avatar.

```
PNLIB_API void PNZeroOutAllAvatar();
```

Remarks

In XY plane.

2.3.135 PNZeroOutPosition

Zero out the avatar position.

```
PNLIB_API void PNZeroOutPosition(int avatarIndex);
```

Parameters

avatarIndex

Avatar index.

Remarks

In XY plane.

```
/*********************************************************
 *                  Action Recognition                   *
 *********************************************************/
```

2.3.136 PNEnableActionRecognition

Enable action recognition.

```
PNLIB_API void PNEnableActionRecognition(BOOL enable);
```

Parameters

enable

BOOL type variable. Set TRUE if enable action recognition, otherwise set FALSE.

Remarks

Default state is disabled.

3 Error code

Code	Message	Remarks
0	No Error.	
1	Init failed.	
2	Load raw data failed. Too many avatar count.	
3	Allocate memory failed.	
4	Buffer is full.	
5	Failed to start thread.	
6	Calibration data file is not exist.	
7	Open file failed.	
8	Unknown rotateOrder.	
9	Constact edit file format error.	
10	Rewind file failed.	
11	Loading file failed: No enough memory.	
12	Fbx format check error.	
13	Fbx format check error.	
14	Raw data version: No version information.	
15	Failed to enter time critical mode.	
16	Not a calibration data file.	
17	Unable to retrieve data descriptions.	
18	Unknown data type.	
19	Open fcd file failed.	

20	Index is out of range.	
21	Incorrect bone index.	
22	You can not set to unknown type.	
23	Function is not implemented.	
24	'AppDataFolder' or 'WorkingFolder' paramenter is illegal.	
25	Buffer is too small.	
26	Invalid pointer.	
27	Can not be less than or equal to 0.	
28	Warning: The defalut forlder of application directory will be used.	
29	Value out of range.	
30	Running mode is error.	
31	Raw file is not exist or open raw file failed.	
100	CreateBvhPlayer FAILED.	
101	Calculate frame count error.	
102	Is not a raw data file.	
103	No data in raw data file.	
104	Read data error.	
105	Load raw data failed. Too many avatar count.	
106	Play raw data failed.	
107	Open raw file failed: wrong frame count.	
108	Runing Mode failed.	
200	Unable to connect to server. Host not present.	
201	Unable to connect to server. /*有错误码传入*/	
202	error un-initting Client.	
103	error re-initting Client.	
300	The current sensor binding is not comfortable with the specified sensor combination mode.	
301	Unsupported mode.	
302	No sensor binded.	
303	The sensor is not bound bone.	
400	The bone index is out of range.	
401	Bone length should be large than 0.	
402	Bone index of BVH output sequence error.	
403	The refered parent node is not exist in bone system.	
500	Avatar index error.	

501	No avatar to deal with this data.	
502	The sensor is not beyond the avatar.	
503	Avatar name is too long	
504	Bone index error.	
600	Bone index error.	
700	Eable constraint edit failed: Constraint editing callback handle not set.\nPlease set handle befor enable constraint edit function.	
701	Wrong frame index.	
702	Wrong level specified. The max level is 10.	
703	Start frameIndex should be less than end frameIndex.	
800	This function is currently not supported in this version.	

4 Appendence

4.1 Calculation callback data

Sample of calculated data output by PNEventCalculatedBinaryDataCallback function.

	Position			Velocity			Rotation				Acceleration			Gyro		
Bone name	X	Y	Z	X	Y	Z	QT_GlobalRawQuat				AT_ModelRaw Data			GY_ModelRaw Data		
第 0 帧：第 0 个角色的数据：							S	X	Y	Z	X	Y	Z	X	Y	Z
Hips	-1.12768	0.511881	-0.0097243	0.0430587	0.17661	-0.0353852	-0.429778	-0.244738	0.521876	0.69503	0.335938	0.945313	0.351563	0.244342	1.58822	0.488684
RightUpLeg	-1.1976	0.433395	0.236806	0.208585	0.00779037	-0.0400262	-0.485756	0.587461	-0.517846	0.388323	-0.0078125	-1.05469	-0.210938	-0.069812	-1.7453	-0.157077
RightLeg	-1.18701	0.449488	0.716166	0.0826626	-0.0252136	-0.0442851	0.256023	-0.231431	0.680769	-0.646122	-0.0820313	-0.960938	0.117188	0.383966	-0.628308	0.296701
RightFoot	-1.34164	0.435341	0.972266	-0.0034509	0.00636844	-0.002375	0.662405	0.173446	0.416738	-0.597907	-0.769531	-0.277344	0.554688	0.052359	-0.104718	-0.052359
LeftUpLeg	-1.13437	0.631112	0.234087	-0.541837	0.136388	-0.199847	-0.376242	0.170877	-0.583707	0.698969	-0.878906	-0.710938	-0.324219	0.855197	-2.16417	1.64058
LeftLeg	-1.14812	0.577846	0.673322	-0.515688	-0.0445268	-0.454843	-0.52536	-0.781522	-0.315507	-0.117061	0.4375	1.05859	-0.464844	1.11699	0.052359	-1.53586
LeftFoot	-1.17612	0.560062	0.958806	-0.340326	-0.210719	-0.70847	0.215539	0.00996922	0.452661	-0.865199	-0.519531	-0.851563	0.875	0.069812	-2.65286	1.39624

1.16935	0.418872	1.08209	1.30897	1.1868	1.60568	-3.00192	-1.22171	-0.139624	RightShoulder
0.104718	-0.715573	-0.244342	-0.191983	1.4486	-3.7873	2.28634	3.92692	0.157077	RightArm
0.977368	0.157077	0.104718	-0.436325	-0.52359	-1.1868	-0.244342	0.296701	0.890103	RightForeArm
0.628906	0.484375	0.328125	0.335938	0.390625	0.0703125	-0.570313	0.816406	-0.4375	RightHand
0.5	-1.125	-0.0351563	0.015625	0.953125	-1.00391	0.617188	1.04297	-0.222656	LeftShoulder
0.703125	-0.0390625	0.804688	-0.320313	-0.265625	-0.492188	0.929688	0.617188	0.953125	LeftArm
0.687415	-0.0605076	-0.76463	-0.499707	-0.843599	-0.713876	0.0663384	-0.100092	-0.070123	LeftForeArm
0.417215	0.243907	-0.364923	-0.408984	-0.383923	0.482584	0.518015	0.033723	0.843245	LeftHand
-0.0570461	0.52083	-0.3318	0.403677	0.303584	0.307707	-0.604061	-0.449569	0.0332307	Head
-0.591749	-0.81585	0.414861	-0.64815	0.220926	-0.403531	-0.601995	-0.886995	-0.531925	Neck
0.0128306	0.0790318	0.207337	0.439725	-0.0714141	-0.0518517	0.220799	0.627343	-0.0386345	Spine
0.0211243	-0.0344646	-0.114271	-0.28238	0.191159	0.432079	0.290374	-0.0951677	0.0563823	Spine1
0.0919324	0.249296	0.252106	0.0888894	-0.135801	-0.43177	-0.304818	0.160922	-0.0618118	Spine2
-0.603523	-0.480021	-0.378716	-0.450763	-0.58644	-0.441431	-0.190611	-0.0142243	-0.772027	Spine3
0.472255	0.34693	0.347166	0.440185	0.612281	0.679129	0.703555	0.767359	0.562099	Spine4
-1.19444	-1.2669	-1.41441	-1.61627	-1.08911	-0.988518	-1.02209	-1.157	-1.14868	Spine5

Neck	Spine3	Spine2	Spine1	Spine					
0.069812	0.069812	0.069812	0.488684	0.488684					
2.3387	2.3387	2.3387	1.58822	1.58822					
-0.296701	-0.296701	-0.296701	0.244342	0.244342					
-0.09375	-0.09375	-0.09375	0.351563	0.351563					
1.10938	1.10938	1.10938	0.945313	0.945313					
0.230469	0.230469	0.230469	0.335938	0.335938					
-0.558399	-0.558399	-0.558399	0.69503	0.69503					
-0.52523	-0.52523	-0.52523	0.521876	0.521876					
0.478	0.478	0.478	-0.244738	-0.244738					
0.428798	0.428798	0.428798	-0.429778	-0.429778					
-0.0339898	-0.0352722	-0.0347064	-0.0379427	-0.0348877					
0.0707543	0.11481	0.167569	0.211958	0.216292					
-0.0061503	0.0285641	0.0692497	0.0736337	0.0699801					
-0.638184	-0.534309	-0.419154	-0.30508	-0.189907					
0.5485	0.537012	0.524117	0.514395	0.508664					
-1.14925	-1.13614	-1.12055	-1.1102	-1.10883					
第 0 帧：第 1 个角色的数据：									
Hips	RightUp Leg	RightLeg	RightFoot						
0.0229068	0.143225	0.159143	0.198163						
-0.0283177	-0.0545468	-0.0440961	-0.186117						
-0.0053774	0.248585	0.72828	0.977035						
-0.0015131	0.00533597	0.00589967	0.00217719						
0.00817126	0.0125785	-0.0093971	-0.0003108						
0.0814736	0.0827218	0.0743723	0.00984643						
-0.730188	-0.389764	-0.152468	-0.926677						
-0.678492	-0.424892	-0.0074615	0.256723						
-0.0299077	0.621061	0.706784	0.199815						
-0.0748922	0.530907	0.690784	-0.188323						
-0.0273438	-0.0625	0.167969	0.296875						
0.960938	0.980469	0.976563	-0.542969						
0.0390625	-0.113281	0.03125	0.78125						
-0.139624	-0.017453	-0.069812	-0.017453						
0.104718	-0.017453	-0.104718	-0.017453						
0.017453	-0.052359	-0.017453	0						

0.052359	-0.052359	0	0.017453	0.296701	0.244342	-0.052359	-0.017453	0.52359
0.052359	0	0	0.069812	0.715573	0.436325	-0.279248	0.279248	0.977368
-0.017453	-0.052359	-0.017453	0.157077	0.052359	0	-0.575949	-0.436325	-0.663214
-0.0078125	-0.171875	0.753906	0.335938	0.0390625	-0.238281	0.1875	0.601563	0.460938
0.996094	0.96875	-0.46875	0.640625	0.992188	0.929688	-0.839844	0.695313	0.515625
-0.0195313	-0.0117188	-0.484375	0.625	-0.183594	-0.226563	-0.332031	-0.285156	-0.996094
-0.61583	-0.628799	0.248154	-0.3162	0.482646	-0.356892	0.101415	0.00303077	0.127969
-0.637107	-0.755276	-0.312477	0.243231	0.406877	-0.234184	-0.380461	-0.144769	-0.467338
-0.333338	0.156892	0.173892	-0.49563	-0.551923	0.72163	0.734861	-0.4206	-0.384261
-0.32212	0.0978882	-0.900315	-0.771515	-0.544898	0.545036	-0.552243	-0.895632	-0.785869
0.0706333	0.0622315	0.00759216	0.0924208	0.0806487	0.0803796	0.062665	0.110786	0.113773
-0.0064345	-0.0289023	-0.0034227	-0.0438292	0.00646447	0.0700584	0.157043	-0.103695	-0.224141
0.00954253	0.00623558	0.00096393	-0.0117713	-0.0463548	-0.115136	-0.090581	-0.0289495	0.0060585
0.247771	0.724818	0.972249	-0.581128	-0.42714	-0.146026	0.0824094	-0.609623	-0.705485
-0.0025672	0.0187155	-0.0987448	-0.0324416	-0.0234402	-0.0310372	-0.035192	0.00133099	-0.0476457
-0.10716	-0.150457	-0.241627	0.165132	0.252586	0.261276	0.277503	-0.0030022	-0.183174

LeftUpLeg	LeftLeg	LeftFoot	RightShoulder	RightArm	RightForeArm	RightHand	LeftShoulder	LeftArm

LeftFore Arm	LeftHand	Head	Neck	Spine3	Spine2	Spine1	Spine		Hips
2.05945	4.43306	-0.069812	-0.017453	-0.017453	-0.017453	0.017453	0.017453		-0.331607
-5.30571	7.13828	-0.017453	0	0	0	0.104718	0.104718		-0.488684
-0.750479	-0.69812	-0.052359	-0.017453	-0.017453	-0.017453	-0.139624	-0.139624		-0.139624
0.117188	-1.98438	-0.15625	-0.0390625	-0.0390625	-0.0390625	0.0390625	0.0390625		0.0507813
-1.00391	0.796875	0.179688	-0.992188	-0.992188	-0.992188	0.960938	0.960938		0.828125
0.765625	1.15625	-0.949219	-0.0664063	-0.0664063	-0.0664063	-0.0273438	-0.0273438		0.15625
-0.676138	0.40043	0.190261	0.670922	0.670922	0.670922	-0.0748922	-0.0748922		0.0392
0.489015	-0.607292	-0.68783	-0.722399	-0.722399	-0.722399	-0.0299077	-0.0299077		0.0392769
-0.479015	0.684353	-0.356507	0.0588153	0.0588153	0.0588153	-0.678492	-0.678492		0.630138
0.272534	0.0504089	-0.60301	-0.156752	-0.156752	-0.156752	-0.730188	-0.730188		0.774516
0.0763995	-0.476624	0.0876011	0.0900384	0.0905671	0.0877916	0.0832096	0.0827922		0.0814205
-0.27885	0.210467	-0.047812	-0.0468947	-0.0470674	-0.0437949	-0.0294445	-0.0150297		-0.239886
0.0915044	0.611969	0.00246469	-0.0060858	-0.0062863	-0.003245	-0.001827	0.00108044		0.227616
-0.903825	-1.08696	-0.768331	-0.635853	-0.531739	-0.416113	-0.30193	-0.186729		-0.0009173
-0.176487	-0.206008	-0.0216944	-0.0213796	-0.0243677	-0.0279335	-0.0301202	-0.0308495		-0.339574
-0.32935	-0.445045	0.11146	0.0858444	0.0695455	0.0516961	0.0358469	0.0258938		1.36637

第 0 帧：第 2 个角色的数据：

RightUpLeg	RightLeg	RightFoot	LeftUpLeg	LeftLeg	LeftFoot	RightShoulder	RightArm	RightForeArm
-1.04718	1.39624	-0.331607	0.139624	-0.052359	0	0.383966	3.31607	1.08209
-2.23398	0.017453	-1.95474	-0.69812	0	0.104718	-0.436325	4.0491	5.93402
-0.34906	-0.767932	-1.69294	0.558496	0.34906	0.209436	0.052359	-0.471231	2.80993
0.03125	-0.410156	-0.0976563	-0.410156	0.0195313	0.816406	0.671875	0.164063	-0.261719
0.574219	-0.230469	-0.0664063	1.23047	0.953125	0.527344	0.640625	1.72266	1.78125
-0.628906	0.277344	0.0234375	0.601563	-0.117188	-0.15625	0.109375	-0.199219	0.296875
0.688599	0.411938	0.767215	0.58303	-0.651846	0.946261	0.2534	0.634015	-0.644476
0.257292	0.865922	0.508738	0.588861	-0.609046	0.259261	-0.0503384	0.498553	-0.729153
-0.59663	-0.205861	-0.229738	0.384015	0.345938	-0.118138	0.418277	-0.399861	-0.0720153
-0.322013	-0.195281	-0.315936	0.407283	0.290716	-0.15312	0.87082	-0.435441	0.218696
-0.112167	-0.278214	-0.387874	0.133361	0.118985	0.0292422	0.154868	0.153348	-0.0238044
-0.485749	-0.727365	-0.841373	-0.126185	-0.0923824	0.00397086	-0.258416	0.315657	1.26152
0.419423	1.0367	1.3573	0.310541	0.405516	0.0423188	0.14268	0.225542	0.183421
0.211346	0.618353	0.920953	0.255212	0.728644	0.978625	-0.576736	-0.443089	-0.179979
-0.436113	-0.45138	-0.447379	-0.397518	-0.473458	-0.648974	-0.453327	-0.437673	-0.409996
1.53263	1.54517	1.52168	1.247	1.25346	1.30257	1.46784	1.59439	1.68709

RightHand	LeftShoulder	LeftArm	LeftForeArm	LeftHand	Head	Neck	Spine3	Spine2
0.418872	0.610855	1.60568	6.57978	7.03356	1.39624	0	0	0
-5.44534	1.1868	0.209436	4.85193	-4.15381	3.63022	-0.34906	-0.34906	-0.34906
-6.02129	0.610855	2.86229	2.37361	1.39624	0.139624	0.191983	0.191983	0.191983
-0.234375	0.375	0.453125	0.425781	0.171875	-0.324219	0.140625	0.140625	0.140625
-1.32031	0.5625	0.925781	0.984375	-2.38672	-0.0429688	1.10156	1.10156	1.10156
0.214844	-0.460938	0.695313	0.726563	-0.171875	-0.730469	-0.0664063	-0.0664063	-0.0664063
0.452876	0.0506923	0.723707	0.95543	-0.238338	0.535215	0.0837845	0.0837845	0.0837845
-0.0646615	-0.205661	0.133892	0.151477	0.0512615	-0.439261	0.120615	0.120615	0.120615
-0.675369	-0.511661	0.50183	0.253154	-0.0850307	-0.646969	0.619553	0.619553	0.619553
0.57847	-0.832685	0.45444	0.0122444	0.966108	-0.319459	0.771112	0.771112	0.771112
-0.156096	0.0538699	0.127809	0.0184852	-0.336317	0.162433	0.0800478	0.0868917	0.0957258
2.08432	-0.381725	-0.779327	-1.21272	-1.593	0.00802859	-0.238472	-0.260095	-0.28999
-0.25516	0.156015	0.558698	1.78431	3.38491	0.411229	0.171148	0.176949	0.187124
0.0276752	-0.565864	-0.451943	-0.277072	-0.202765	-0.744652	-0.610613	-0.511538	-0.401722
-0.367522	-0.491559	-0.566189	-0.734718	-0.933945	-0.482044	-0.48266	-0.447499	-0.407788
1.7718	1.29816	1.13305	1.04227	0.984914	1.38717	1.38286	1.37762	1.36997

Spine1	Spine		Hips	RightUpLeg	RightLeg	RightFoot	LeftUpLeg	LeftLeg	LeftFoot
-0.331607	-0.331607		0.802838	-0.471231	0.157077	-0.017453	1.04718	-1.69294	1.32643
-0.488684	-0.488684		2.02455	-1.46605	-0.942462	-0.052359	-3.76985	1.97219	-2.42597
-0.139624	-0.139624		0.418872	0.209436	0.191983	-0.017453	0.34906	0.802838	-0.017453
0.0507813	0.0507813		0.347656	-0.191406	0.203125	0.578125	-0.0898438	-0.867188	0.417969
0.828125	0.828125		0.917969	-1.09766	-0.949219	-0.273438	-0.839844	0.84375	-0.285156
0.15625	0.15625		0.449219	-0.0507813	-0.101563	-0.738281	-0.675781	0.355469	-0.101563
0.0392	0.0392		0.678953	0.375384	-0.647692	-0.596892	0.685999	-0.112584	-0.859815
0.0392769	0.0392769		0.515323	-0.500646	0.673784	0.416307	-0.560307	-0.348215	0.443107
0.630138	0.630138		-0.263154	0.599322	-0.232646	0.171723	0.190969	-0.778122	-0.0160615
0.774516	0.774516		-0.451933	-0.499281	0.2691	0.664039	-0.423107	-0.510514	0.253269
0.0926124	0.0854883		-0.0548433	-0.0333884	-0.0409642	-0.003182	-0.173434	-0.434936	-0.636259
-0.294268	-0.272988		0.209192	0.101043	0.124602	0.0163986	0.167316	0.0150459	-0.170409
0.199622	0.217584		0.0164205	0.160565	0.0108044	-0.0080132	-0.474172	-0.334641	-0.024846
-0.294399	-0.181695		-0.0082027	0.238688	0.717669	0.973525	0.229764	0.658186	0.936928
-0.372281	-0.348912		0.514508	0.43141	0.448757	0.435781	0.633061	0.575546	0.552246
1.36589	1.36336		-1.12744	-1.1917	-1.18634	-1.34124	-1.15447	-1.16476	-1.18201

第 1 帧：第 0 个角色的数据：

RightShoulder	RightArm	RightForeArm	RightHand	LeftShoulder	LeftArm	LeftForeArm	LeftHand	Head
1.55332	0.942462	1.83256	2.3038	1.36133	0.994821	-1.91983	0.383966	-0.052359
1.1868	-0.907556	-0.802838	0.750479	2.14672	-4.20617	2.53068	3.76985	0.069812
1.02973	0.52359	0	-0.331607	-0.994821	-1.16935	0.383966	-1.36133	0.994821
0.582031	0.507813	0.203125	0.242188	0.359375	0.152344	-0.527344	1.36328	-0.410156
0.542969	-1.14063	-0.242188	0.304688	1.05469	-1.125	0.953125	1.51953	-0.328125
0.886719	-0.117188	0.796875	-0.367188	-0.191406	-0.246094	0.664063	0.503906	0.890625
0.665661	-0.0785538	-0.75083	-0.519784	-0.833907	-0.732738	0.0664922	-0.1256	-0.0818768
0.4224	0.248892	-0.360338	-0.42403	-0.372769	0.515907	0.468461	-0.0312615	0.840707
-0.0657538	0.517584	-0.347723	0.397492	0.317369	0.276738	-0.627538	-0.429892	0.0262923
-0.611701	-0.814872	0.430729	-0.62614	0.254848	-0.346957	-0.618342	-0.893571	-0.534645
-0.0302582	0.00866896	0.237106	0.628188	-0.0304669	0.00868681	0.124958	0.253843	-0.0544925
-0.0311886	-0.118405	-0.218387	-0.407519	0.223051	0.397986	0.225013	0.0304947	0.024951
0.162203	0.470458	0.535528	0.340843	-0.219382	-0.496691	-0.421177	-0.283583	-0.0587974
-0.600294	-0.474665	-0.367262	-0.428716	-0.585303	-0.439186	-0.1804	0.00768575	-0.770746
0.46698	0.338933	0.336211	0.423063	0.614563	0.690215	0.709174	0.761321	0.557574
-1.19196	-1.25611	-1.40216	-1.61	-1.09783	-1.00876	-1.0388	-1.16185	-1.15311

Neck	-1.1519	0.54589	-0.636616	-0.0245552	0.0684398	-0.0462129	0.428798	0.478	-0.52523	-0.558399	0.230469	1.10938	-0.09375	-0.296701	2.3387	0.069812
Spine3	-1.13763	0.53602	-0.532731	0.00885563	0.109064	-0.0472358	0.428798	0.478	-0.52523	-0.558399	0.230469	1.10938	-0.09375	-0.296701	2.3387	0.069812
Spine2	-1.12073	0.524995	-0.417556	0.0456395	0.150926	-0.0486819	0.428798	0.478	-0.52523	-0.558399	0.230469	1.10938	-0.09375	-0.296701	2.3387	0.069812
Spine1	-1.10968	0.51673	-0.303551	0.050058	0.191397	-0.052942	-0.451933	-0.263154	0.515323	0.678953	0.449219	0.917969	0.347656	0.418872	2.02455	0.802838
Spine	-1.10794	0.511821	-0.18833	0.0461393	0.216907	-0.051489	-0.451933	-0.263154	0.515323	0.678953	0.449219	0.917969	0.347656	0.418872	2.02455	0.802838

4.2 BVH callback data

Sample of BVH data output by PNEventBVHBinaryDataBoardcastCallback function.

		Position			Rotation		
		X	Y	Z	X	Y	Z
	第 0 个角色第 0 帧：						
	reference	0	0	0	0	0	0
1	Hips	-112.16	103.06	50.12	-10.89	-48.58	-11.48
2	RightUpLeg	-10.45	-0.83	-4.99	9.3	-10.26	6.05
3	RightLeg	-2.72	-47.71	2.18	-2.66	-19.97	0.17
4	RightFoot	-2.19	-46.3	2.12	-0.06	-17.36	-6.03
5	LeftUpLeg	5.93	-1.62	2.77	-16.03	-5.94	-6.3
6	LeftLeg	2.89	-45.85	-1.55	50.58	14.76	-1.13
7	LeftFoot	-0.08	-47.49	-4.65	-0.12	-9.04	-2.39
8	Spine	0.41	13.85	0.49	10.35	-0.77	2.24
9	Spine1	-0.4	11.29	-0.18	2.61	-0.17	0.54
10	Spine2	-0.82	11.72	-0.28	4.32	-0.22	0.9
11	Spine3	-0.99	11.24	-0.37	-0.47	-0.89	-0.17

12	Neck	-0.85	12.05	-0.16	-0.47	-0.89	-0.17
13	Head	-0.99	8.92	0.47	-10.49	-15.63	-4.48
14	RightShoulder	-2.26	7.08	-3.72	-1.24	-4.41	-4.02
15	RightArm	-12.91	-1.19	-5.17	-12.79	-26.23	69.36
16	RightForeArm	-35.31	-0.18	-4.4	-153.06	77.52	164.48
17	RightHand	-22.99	2.68	-10.28	-28.08	12.36	3.48
18	RightHandThumb1	-2.7	0.21	3.39	0	0	0
19	RightHandThumb2	-2.75	-0.64	2.83	0	0	0
20	RightHandThumb3	-2.13	-0.81	1.59	0	0	0
21	RightInHandIndex	-3.5	0.55	2.15	0	0	0
22	RightHandIndex1	-5.67	-0.1	1.08	0	0	0
23	RightHandIndex2	-3.92	-0.19	0.2	0	0	0
24	RightHandIndex3	-2.22	-0.14	-0.08	0	0	0
25	RightInHandMiddle	-3.67	0.56	0.82	0	0	0
26	RightHandMiddle1	-5.62	-0.09	0.34	0	0	0
27	RightHandMiddle2	-4.27	-0.29	-0.2	0	0	0
28	RightHandMiddle3	-2.67	-0.21	-0.24	0	0	0
29	RightInHandRing	-3.65	0.59	-0.14	0	0	0
30	RightHandRing1	-5	-0.02	-0.52	0	0	0
31	RightHandRing2	-3.65	-0.29	-0.74	0	0	0
32	RightHandRing3	-2.55	-0.19	-0.44	0	0	0
33	RightInHandPinky	-3.43	0.51	-1.3	0	0	0
34	RightHandPinky1	-4.49	-0.02	-1.18	0	0	0
35	RightHandPinky2	-2.85	-0.16	-0.9	0	0	0
36	RightHandPinky3	-1.77	-0.14	-0.66	0	0	0
37	LeftShoulder	1.38	8.81	3.7	-7.94	-1.28	-4.28
38	LeftArm	16.68	0.05	6.71	7.71	12.37	-74.31
39	LeftForeArm	30.32	-1.16	-1.17	-8.85	-36.74	-33.58
40	LeftHand	27.34	-3.21	-6.64	8.77	-13.61	9.31
41	LeftHandThumb1	2.7	0.21	3.39	0	0	0
42	LeftHandThumb2	2.75	-0.64	2.83	0	0	0
43	LeftHandThumb3	2.13	-0.81	1.59	0	0	0
44	LeftInHandIndex	3.5	0.55	2.15	0	0	0
45	LeftHandIndex1	5.67	-0.1	1.08	0	0	0
46	LeftHandIndex2	3.92	-0.19	0.2	0	0	0
47	LeftHandIndex3	2.22	-0.14	-0.08	0	0	0
48	LeftInHandMiddle	3.67	0.56	0.82	0	0	0
49	LeftHandMiddle1	5.62	-0.09	0.34	0	0	0
50	LeftHandMiddle2	4.27	-0.29	-0.2	0	0	0
51	LeftHandMiddle3	2.67	-0.21	-0.24	0	0	0
52	LeftInHandRing	3.65	0.59	-0.14	0	0	0
53	LeftHandRing1	5	-0.02	-0.52	0	0	0
54	LeftHandRing2	3.65	-0.29	-0.74	0	0	0

55	LeftHandRing3	2.55	-0.19	-0.44	0	0	0
56	LeftInHandPinky	3.43	0.51	-1.3	0	0	0
57	LeftHandPinky1	4.49	-0.02	-1.18	0	0	0
58	LeftHandPinky2	2.85	-0.16	-0.9	0	0	0
59	LeftHandPinky3	1.77	-0.14	-0.66	0	0	0
	第 1 个角色第 0 帧：						
	reference	0	0	0	0	0	0
1	Hips	2.1	103.55	-4.47	178.79	-13.49	179.6
2	RightUpLeg	-11.56	-1.48	0.57	-1.99	-31.39	-2.14
3	RightLeg	0.08	-48.03	0.29	2.45	-0.38	-0.59
4	RightFoot	-0.39	-46.88	-1.08	-2.84	4.97	2
5	LeftUpLeg	11.42	-1.31	-0.13	-2.65	2.98	4.43
6	LeftLeg	-0.13	-48.04	0.54	4.48	10.79	2.81
7	LeftFoot	0.43	-47.42	-1.15	-4.6	0.43	0.2
8	Spine	-0.11	13.88	0.13	-2.6	-2.42	4.89
9	Spine1	0.02	11.32	0.12	-0.64	-0.62	1.23
10	Spine2	0.12	11.78	0.11	-1.02	-1.07	2.05
11	Spine3	0.14	11.31	0.12	0.11	-0.74	0.17
12	Neck	0.13	12.1	0.11	0.11	-0.74	0.17
13	Head	0.37	8.99	0.04	2.79	-13.35	3.48
14	RightShoulder	-3.42	8.18	0.18	2.72	-4.65	-1.73
15	RightArm	-17.06	0.11	-0.15	9.55	22.98	83.55
16	RightForeArm	-29.12	-0.92	-0.23	-2.18	15.84	-9.7
17	RightHand	-27.54	-0.13	-1.14	31.85	-14.06	15.81
18	RightHandThumb1	-2.7	0.21	3.39	0	0	0
19	RightHandThumb2	-2.75	-0.64	2.83	0	0	0
20	RightHandThumb3	-2.13	-0.81	1.59	0	0	0
21	RightInHandIndex	-3.5	0.55	2.15	0	0	0
22	RightHandIndex1	-5.67	-0.1	1.08	0	0	0
23	RightHandIndex2	-3.92	-0.19	0.2	0	0	0
24	RightHandIndex3	-2.22	-0.14	-0.08	0	0	0
25	RightInHandMiddle	-3.67	0.56	0.82	0	0	0
26	RightHandMiddle1	-5.62	-0.09	0.34	0	0	0
27	RightHandMiddle2	-4.27	-0.29	-0.2	0	0	0
28	RightHandMiddle3	-2.67	-0.21	-0.24	0	0	0
29	RightInHandRing	-3.65	0.59	-0.14	0	0	0
30	RightHandRing1	-5	-0.02	-0.52	0	0	0
31	RightHandRing2	-3.65	-0.29	-0.74	0	0	0
32	RightHandRing3	-2.55	-0.19	-0.44	0	0	0
33	RightInHandPinky	-3.43	0.51	-1.3	0	0	0
34	RightHandPinky1	-4.49	-0.02	-1.18	0	0	0
35	RightHandPinky2	-2.85	-0.16	-0.9	0	0	0

36	RightHandPinky3	-1.77	-0.14	-0.66	0	0	0
37	LeftShoulder	3.56	7.98	-0.07	-5.87	4.52	4.25
38	LeftArm	17.96	-0.25	-0.54	-90.25	-27.37	-46.97
39	LeftForeArm	28.83	0.69	0.14	-53.95	-19.88	-16.19
40	LeftHand	27.84	-1.13	2.87	-21.76	-30.69	31.6
41	LeftHandThumb1	2.7	0.21	3.39	0	0	0
42	LeftHandThumb2	2.75	-0.64	2.83	0	0	0
43	LeftHandThumb3	2.13	-0.81	1.59	0	0	0
44	LeftInHandIndex	3.5	0.55	2.15	0	0	0
45	LeftHandIndex1	5.67	-0.1	1.08	0	0	0
46	LeftHandIndex2	3.92	-0.19	0.2	0	0	0
47	LeftHandIndex3	2.22	-0.14	-0.08	0	0	0
48	LeftInHandMiddle	3.67	0.56	0.82	0	0	0
49	LeftHandMiddle1	5.62	-0.09	0.34	0	0	0
50	LeftHandMiddle2	4.27	-0.29	-0.2	0	0	0
51	LeftHandMiddle3	2.67	-0.21	-0.24	0	0	0
52	LeftInHandRing	3.65	0.59	-0.14	0	0	0
53	LeftHandRing1	5	-0.02	-0.52	0	0	0
54	LeftHandRing2	3.65	-0.29	-0.74	0	0	0
55	LeftHandRing3	2.55	-0.19	-0.44	0	0	0
56	LeftInHandPinky	3.43	0.51	-1.3	0	0	0
57	LeftHandPinky1	4.49	-0.02	-1.18	0	0	0
58	LeftHandPinky2	2.85	-0.16	-0.9	0	0	0
59	LeftHandPinky3	1.77	-0.14	-0.66	0	0	0